Computer Science and Information Systems 12(1):323–348 DOI: 10.2298/CSIS140301002R

Identifying Common Activities in the Graphical User Interface Development Process and their integration into the Software-System Development Life Cycle

Laura C. Rodriguez-Martinez[1], Hector A. Duran-Limon[2], Ricardo Mendoza-González[3], and Jaime Muñoz[4],

[1] Institute of Technology of Aguascalientes, Av. Adolfo López Mateos 1801 Ote., Bona Gens, CP 20256, Aguascalientes, Ags., Mexico
lrodriguez@mail.ita.mx
[2] CUCEA, University of Guadalajara, Periférico Norte 799, Zapopan, Jalisco, Mexico
hduran@cucea.udg.mx
[3] Postgraduate Studies Universidad Politecnica de Aguascalientes, Calle Paseo San Gerardo 207, San Gerardo, CP 20342, Aguascalientes, Ags., Mexico
mendozagric@mail.ita.mx
[4] Autonomous University of Aguascalientes, Av. Universidad 940, Ciudad Universitaria, CP 20131, Aguascalientes, Ags., Mexico
jmunozar@correo.uaa.mx

Abstract. We identify a suite of activities in the development process of Graphical User Interfaces (GUI) and include them as part of an approach to a generic model for the GUI Development Process (GDP). This work contributes with: (1) the identification of common activities of previous GDPs, (2) the definition of an approach to a generic GDP limited to its phases and activities, and (3) the integration of such a generic GDP with any software-system development life cycle (SDLC), illustrated with the Spiral SDLC. We show this work is useful by a) highlighting the advantages of our proposal over other methodologies for GDP in Human-Computer Interaction (HCI), b) showing one example of the integration of the GDP into a SDLC, and c) showing the usefulness of our approach in a case example.

Keywords: System Development Life-Cycle, Software Engineering, Graphical User Interface Development Process, Human-Computer Interaction.

1. Introduction

The research areas of Software Engineering (SE) and Human-Computer Interaction (HCI) are disjoint areas with respect to the processes that involve each one of these areas. That is, the former involves a system life-cycle (SDLC) process whereas the latter is related to a Graphical User Interface Development Process (GDP). Such a separation of these bodies of knowledge, make it hinder to reach a real impact of HCI on the products of the SDLC process since a software engineer needs to easily incorporate a GDP into a SDLC, and the usability practitioner needs to easily participate along all the

SDLC process. Where usability practitioner is the person who applies the knowledge of HCI to design usability GUIs.

The study of a SDLC implies a process perspective, then, if we analyse the GUI design in a process perspective, we can include such a GDP as a natural part of a SDLC [8]. In HCI there are several proposals to develop Graphical User Interfaces (GUIs), in this sense, it is clear that all of them agree on the importance of having a methodology (defined process, methods and techniques) to develop GUIs [9], [1], [11], [18], but also, all of them –in a general way– lack of a process that ensures a systemic way to work. These GDPs are explained by means of the definition of one or several of the following large characteristics: *general characteristic 1 (GC1)*, *general characteristic 2 (GC2)* and *general characteristic 3 (GC3)*. GC1 [2], [7] is defined as phases without details and it highlights the way to iterate. GC2 [6], [9] is either a suite of process deliverables or a model of deliverables. Lastly, GC3 [22] is a suite of methods and techniques that can be applied through the same process. On the other hand, for a well-defined process in SE there is a metamodel that defines the process model of the software development in which it is indicated that a process has a number of phases, activities, deliverable artifacts, roles, and tools (methods, techniques and technology).

Therefore we have that: (1) current proposals about GUI development do not define the GDP in an explicit way, and neither indicate how the GDP can be included in a SDLC [18], [2], [15]; (2) methodologies approaches in SE do not detail the development process of GUIs [9], [1], [6]. Also, such an integration requires at minimum the experience of GUI development coming from HCI and the methodological experience of SE.

Since 1985, several authors have recognised the "*need to view human-computer interface management as an integral part of the software process*" [11][1][9][25][8]. Therefore, it has been recognized that the development of interfaces should not be carried out in isolation from the development of the rest of the application system, as if the interface were an "add-on" part of a system. Based on this argument, Hartson & Hix proposes in [11] a methodology for interactive system development that is called "star" life cycle, which highlights the interface development process as an integral part of a system development process. However, such a proposal can only be used as a process for GUI development since this methodology represents the implementation of the software system as a "black box". In a similar way, in SE, Boehm includes in his "Spiral model" of software development [24], user involvement, prototyping, and iterative design strategies that Lewis and Gould suggest in [7]. Similarly, User Centered Design (UCD) emerged from HCI. UCD is software design methodology for developers and designers, that essentially focuses on reaching applications that meet user needs [10], [14] . Regarding this last issue, there are several contributions that addressed their efforts to join UCD to SDLC e.g. [25], [8], [3]. Nevertheless, these proposals of UCE are not specified enough slightly attending important software development details from SE.

In the last years (2003, 2009, 2012) there have been some efforts in the HCI [9], [1] and in the SE community [21] to integrate both the GUI and system development processes. In practice, despite all efforts, such an integration of GDPs into SDLCs normally requires a customization, and depending on the specific process model followed, resources devoted to the design of a user interface are concentrated at different stages of a project [25]. Also, some researches believe that any process model

can apply the HCI process either at the first or in the final stage. This approach is called a dichotomy of User Interface first (UIF) versus User Interface later (UIL) [25]. However, such all above efforts only consider specific SDLCs with a specific integrated GDP. Further more, such efforts still present the implementation of the software system as a "black box" (i.e. they do not indicate how the activities of the GDP can be intercalated with the activities of a SDLC process).

In this paper, we present an approach to a generic well-defined GDP. Our contribution focuses on defining phases and activities of such a GDP, and on specifying what and when the activities of the GDP need to be executed in a SDLC. The GDP's deliverable artifacts, roles and associated tools are not defined, and are regarded as future work. The main contributions of this paper involve: (1) a process perspective approach to a generic GDP that explicitly defines its phases and activities; (2) our GDP enables its integration into an arbitrary SDLC; and (3) our GDP explicitly defines how the activities of the GDP are intercalated with the activities of an SDLC (as opposed to the approaches in which a GDP is represented as a "black-box" when integrating it with a SDLC or a SDLC is represented as a "black-box" and then integrated into a GDP).

This paper is structured as follows. Section 2 presents an analysis of the strengths and weaknesses of previous GDPs. In Section 3, we introduce our proposal involving an improved GDP and elaborate on how it can be integrated into a SDLC. A case example is described in Section 4 to show how our proposed GDP can be used independently of any SDLC. Our approach is evaluated in Section 5. Finally, some concluding remarks are given in Section 6.

2. An Analysis of Strengths and Weaknesses of Previous GDPs

We carried out a literature review involving the period 1985 to 2009 because in those years the research efforts tended to develop proposals with a higher specification level for GDPs. From 2009 onwards, research proposals were more focused on illustrative "black-boxes" fashion models omitting a number of details and descriptions [6]. Although such models serve as a visual tool, we believe it is essential to describe the activities and interaction of a GDP in the SDLC.

2.1. Brief description of previous GDPs

We have reviewed ten selected GDPs. Such selected GDPs are named -as H1..H10- and described from a process point of view, as follows:

H1. **"Designing for Usability: Key Principles and What Designers Think" Gould & Lewis, 1985 [7].** It proposes two phases with no clear separation line between them as follows: (1) Initial Design Phase: this phase defines the next activities: (a) Preliminary Specification of the User Interface, (b) Collect Critical Information About Users, (c) Develop Behavioral Goals, and (d) Organize the Work. (2) Iterate Development Phase: there is an iterative process that ensures the next usability pints: (a) Early focus on user, (b) Iterative design, and (c) Empirical Measurement. It does not properly describe the phases-activities.

H2. **"Designing for Designers: Analysis of Design Practice en the Real World" Rosson, Maass, & Kellogg, 1987 [22]**. It focus on tool development for its use by GUI designers. It proposes an iterative process, and a combination of two perspectives: process and tools. The author considers that the use of adequate tools can provide a great impact over the quality of the designed interactive system. The proposal documents the results of qualitative interviews to GUI designers about their experiences given support to ideas like "an iterative approach in the general process of design and user" between others. However, there was not reported a consensus on: the GUI design process. H2 proposes the following stages: (1) design phase, (2) an implementation phase, and (3) one short evaluation phase. Such phases are not properly described.

H3. **"Human-Computer Interface Development: Concepts and Systems for its Management" Hartson & Hix, 1989 [11].** It proposes the utilization of prototyping, principles of usability, and a star life cycle for human-computer interface development with the following activities: (1) Implementation, (2) Task analysis/functional analysis, (3) Prototyping, Conceptual design/formal, (4) Design Representation, (5) Requirements Specification, and (6) Evaluation. Star lifecycle does not propose a specific order to conduct its activities. The unique restriction is that, after each activity execution among the activities 1 to 5 must be performed the activity 6. It does not describe the activities proposed.

H4. **"The Usability Engineering Life Cycle" Nielsen, 1992 [18].** It proposes the Usability Engineering (UE) [18]. Author considers that the process of UE can be easily incorporated into an iterative-incremental development process, as ordered steps. His proposal of UE process has the following phases: (1) Predesign, (2) Design, and (3) Postdesign. H4 describes such three large phases and also considers the following critical aspects: user evaluation, prototyping and iterative design.

H5. **"Iterative Methodology and Designer Training in Human-Computer Interface Design" Bailey, 1993 [2].** It inserts two particular results of GUI design process: (1) use of methods and tools, and (2) training and knowledge of the self-designers. This proposal also considers an iterative design process, the use of prototyping, and usability guidelines. This proposal does not properly describe the implicit and supported phases of Design, Prototyping and Evaluation.

H6. **"Usability Engineering Turns 10" Butler, 1996 [4].** It considers the UE basic-cycle (analysis-design-construction-evaluation) and the iterative prototyping. The iterative cycle ends when the evaluation reaches satisfactory results. This proposal describe the four phases proposed.

H7. **"Design Methodology and Design Practice" Lowgren & Stolterman, 1999) [15].** It emphasises the importance of using methods for GUI design. Several explicit recommendations are reported for its proposed phases: Detailed Design, Implementation, and Evaluation; and in an implicit way the phase of Design Contextualization, in which the designer contextualizes the need of the users, the organization environment and the functional analysis. This is a short study and the author does not explain with details these phases.

H8. "Guía de actuación en el desarrollo de interfaces de usuario según la metodología centrada en el usuario INTEGRAM" Losada, López, & Martínez, 2004 [13]. It is based on three design processes of HCI: (1) ISO 13-407, (2) Greemberg 1996, and (3) star life cycle for human-computer interface development [22]. H8 proposes several methodological recommendations on how to insert the usability and strong iteration. It

proposes the stages: (1) analysis (task analysis/functional analysis, requirements specification), (2) design (conceptual design/formal design), (3) prototyping, (4) interface construction (implementation), and (5) evaluation. It does not properly provide a description neither an order to the stages proposed.

H9. **"Diseño de Sistemas Interactivos Centrados en el Usuario" Granollers, Lorés, & Cañas, 2005 [9].** It proposes the use of UCD proposing the stages and their descriptions: (1) Needs Analysis, (2) User and Task Analysis, (3) Functional Analysis, (4) Requirements Analysis, (5) Design, (6) Prototype, (7) Implementation, (8) Evaluation.

H10. "A Survey on HCI in the Software Development Life Cycle: from Practitioner's Perspective" Abd Majit, Md Noor, Wan Adnan & Mansor, 2009 [1]. It emphasises the use of guidelines and principles in a normal SDLC. This approach is called Human-centered system Development Life Cycle (HCSDCL) and proposes the use of the common phases of a SDLC: (1) planning, (2) analysis, (3) design and (4) implementation; and aggregate into them, tasks of HCI. Its main limitation is that does not provide detailed relations among such SDLC activities and the HCI guidelines and principles, and it does not provide descriptions of phases or tasks.

2.2. Normalization of GDP's Activities

We used, in a general way, a conceptual research method [16] to perform an analysis of previous methods. As the basis of this analysis we took the normalized table defined in [2] where the ten GDPs are reviewed from the HCI literature. We analysed such GDP's in order to find their weaknesses and their strengths. We defined a *weakness* of a GDP from the methodological point of view as a process that is not well-defined when such a process does not include one or more of the following aspects: all phases and activities, a complete description of phases and activities as well as the specification of how the process iterates. *Strengths* were defined as the phases and activities that are included in most of the reviewed approaches. We have limited the scope of our analysis to cover weaknesses regarding only the definition of phases and activities.

The analysis of the ten GDPs allows for a better understanding on how we can integrate the development process of GUIs to a SDLC. We also used the analysis of GDP's presented in [21] that identifies, sorts, and normalises the activities of the ten GDPs (see Table 1). Table 1 is organised as follows. Both, columns 1 and 2 encompass generic phases at different levels of abstraction. The term *Macro-Phases* refers to phases at higher level and the term *Phases* refers to phases at lower level. Finally, columns 3 to 12 present the activities of each GDP through the macro-phases and phases (c.f. [21]).

Table 1. Normalised activities of GDPs (Quoted from [21], p. 22)

<table>
<tr><th>Macro-phase</th><th>Phase</th><th>H1
Gould and Lewis, 1985 [7]</th><th>H2
Rosson, 1987 [22]</th><th>H3
Hartson and Hix, 1989 [11]</th><th>H4
Nielsen, 1992 [18]</th><th>H5
Bailey, 1993 [2]</th><th>H6
Butler, 1996 [4]</th><th>H7
Lowgren and Stolterman, 1999 [15]</th><th>H8
Losada, 2004 [13]</th><th>H9
Granollers, 2005 [9]</th><th>H10
Abd Majid, 2009 [1]</th></tr>
<tr><td rowspan="4">Definition</td><td rowspan="4">Requirements</td><td>Organize the Work.</td><td>Other Activities (i.e. marketing activities)</td><td>---</td><td>---</td><td>---</td><td>---</td><td>---</td><td>---</td><td>---</td><td>Project selection and planing</td></tr>
<tr><td>Preliminary Specification of the User Interface.</td><td>---</td><td>---</td><td rowspan="3">Predesign</td><td>---</td><td rowspan="3">Analysis</td><td rowspan="3">Design Contextuali-zation</td><td rowspan="3">Analysis: task analysis / functional analysis, requirements specification</td><td>Analysis of Needs</td><td rowspan="3">Analysis: Requirements Determination, User Needs Test, Context Analysis, User Analysis, Task Analysis, Formative Evaluation and Alertnative Selection</td></tr>
<tr><td>Collect Critical Information About Users.</td><td>---</td><td>Requirements Specification</td><td>---</td><td>Requirements Analysis</td></tr>
<tr><td>Develop Behavioral Goals.</td><td>---</td><td>Functional Analysis of Tasks</td><td>---</td><td>User Analysis and Task Analysis + Functional Analysis</td></tr>
<tr><td rowspan="3">Development</td><td>Design</td><td rowspan="3">Iterative Development Phase</td><td>Design</td><td>Formal Conceptual Design</td><td rowspan="2">Design</td><td>Design</td><td>Design</td><td>Detailed Design</td><td>Design: conceptual design/formal design</td><td>Design</td><td>Design</td></tr>
<tr><td>Construction</td><td rowspan="2">Implementa-tion</td><td>Prototyping</td><td rowspan="2">Prototyping</td><td rowspan="2">Construction</td><td rowspan="2">Implementa-tion</td><td rowspan="2">Prototyping</td><td rowspan="2">Prototype</td><td rowspan="3">Implementa-tion: coding, formative evaluation</td></tr>
<tr><td>Evolution</td><td>Operation</td><td>Interface Construction</td></tr>
<tr><td></td><td></td><td>Evaluation</td><td>Evaluation</td><td>Postdesign</td><td>Evaluation</td><td>Evaluation</td><td>Evaluation</td><td>Evaluation</td><td>Evaluation</td></tr>
</table>

The authors in [21] normalised the activities based on both the Macro-Phases (Definition, Development and Evolution), and the Phases (Requirements, Design, Construction and Operation) [20]. Such comparison framework [20] has its base on thirteen relevant traditional and contemporary SDLC models, and it is useful to map the variant structure of activities, phases and/or stages of each GDP without loss of a coherent order [21].

We have identified strengths and weakness of the ten selected GDPs according to two points of view: ***PV1*** **-** the normalisation of the activities of the GDPs presented in Table 1; and ***PV2*** **-** a minimum characterisation of the "definition" of the GDPs. Below we present the strengths identified according to PV1 and the strengths and weakness identified according to PV2.

2.3. Strengths of the Ten Selected GDPs regarding the Point of View PV1

The performed analysis revealed similar phases/activities –that we consider as activities in Table 1– among the GDPs, which we identified as "strengths" of the GDPs, as said earlier. We believe that such phases/activities can be used to define a generic GDP. Then to propose the activities of our GDP we generalised the previous normalized activities (that are shown in the column 3 of the Table 2), from phases/activities of the columns 2 to 12 of the ten GDPs.

Table 2. Normalised and generalised activities of GDPs (modified from [21], p. 22)

Macro-phase	Phase	Generic GDP activities	H1 Gould and Lewis, 1985 [7]	H2 Rosson, 1987 [22]	H3 Hartson and Hix, 1989 [11]	H4 Nielsen, 1992 [18]	H5 Bailey, 1993 [2]	H6 Butler, 1996 [4]	H7 Lowgren and Stolterman, 1999 [15]	H8 Losada, 2004 [13]	H9 Granollers, 2005 [9]	H10 Abd Majid, 2009 [1]
Definition	Requirements	Project Selection Marketing Activities Organize the work	Organize the Work.	Other Activities (i.e. marketing activities)	---	---	---	---	---	---	---	Project selection and planning
		Analysis of Needs User Analysis Task Analysis Functional Analysis Requirements Specification Preliminary Specification of UI	Preliminary Specification of the User Interface.	---	---	Predesign	---	Analysis	Design Contextuali-zation	Analysis: task analysis / functional analysis, requirements specification	Analysis of Needs	Analysis: Requirements Determination, User Needs Test, Context Analysis, User Analysis, Task Analysis, Formative Evaluation and Alertnative Selection
			Collect Critical Information About Users.	---	Requirements Specification		---				Requirements Analysis	
			Develop Behavioral Goals.	---	Functional Analysis of Tasks		---				User Analysis and Task Analysis + Functional Analysis	
Development	Design	Conceptual Design Formal Design Detailed Design	Iterative Development Phase	Design	Formal Conceptual Design	Design	Design	Design	Detailed Design	Design: conceptual design/formal design	Design	Design
	Construction	Prototyping (includes implementation of operational prototypes and proof of versión) Deployment		Implementa-tion	Prototyping Interface Construction		Prototyping	Construction	Implementa-tion	Prototyping	Prototype	Implementation: coding, formative evaluation
				Evaluation	Evaluation	Postdesign	Evaluation	Evaluation	Evaluation	Evaluation	Evaluation	
Evolution	Operation	Operation Evaluation Evolution										

Also, we propose the phases as follows. As a first step, we included the four phases in H6 (Analysis, Design, Construction and Evaluation) that generalises most phases/activities of the GDPs. Next, we included the Predesign phase (that is interpreted as a previous design), which is present in H4. The Planning phase, which is present in

H1, H2, and H10 was also included to make it a complete process. Table 3 shows the resulting selection of phases, which are the following: Planning, Analysis, Predesign, Design, Construction, and Evolution (the name of the latter phase is changed from "Evaluation" to "Evolution"). Others approaches apart from the ten proposals include details of some phases such as "Analysis of Needs", "Requirements Analysis" and "User Analysis"; e.g. GDP H9 includes details of phase "Analysis". We considered that these details could be represented by the activities in the phases of a generic GDP.

2.4. Strengths of the Ten Selected GDPs regarding the Point of View PV2

We found that GDPs are defined in different ways. However, most GDP are not described as well-defined processes. We carried out an analysis about a characterization of the GDPs' definition.

Oktaba [19] states that a software process is defined in terms of phases, activities, deliverable artifacts, roles and agents. In this paper, the scope of the analysis is limited to the core definition of a *process*, which includes phases, activities, their descriptions, and the iterative mode of the process. Based on this limited scope, in order to identify the weakness and strengths of the ten GDPs we defined and used four characteristics that we identified in the ten GDPs:

- ***PV2-1.*** Definition of Phases/Activities. These characteristics ensure that the process has defined at least a list of phases and/or activities.
- ***PV2-2.*** Integration into a SDLC. This characteristic refers to the capability of enabling the integration of the GDP into a SDLC. We consider this characteristic is a strength since H3 (since 1989), H4 and H7 do not conceive the possibility of separating a GDP from a SDLC. Even the newest approaches (e.g. H8, H9, and H10) consider the integration of a GDP into a SDLC.
- ***PV2-3.*** Detailed Description of Phases/Activities. This characteristic indicates that the process has a description of phases and/or activities, given that a detailed description of them is provided as an explicative text. In the case of phases, the description is still considered as detailed if the activities that define the phase are defined.
- ***PV2-4.*** Define the iterative process. This characteristic explicitly defines how the process iterates.

Following, we can identify specific strengths and weaknesses involving characteristic ***PV2-1***. H1 defines phases and activities in detail, but this GDP does not describe them. H2 and H3 indicates phases/activities that are not described. Both, H4 and H6 define and describe phases, but none of them define activities. H5 does not define phases/activities, but it is possible to infer its activities. H7 and H8 define phases without giving a detailed description, furthermore, none of them define activities. H9 does not define phases, although it indicates them as a black box. H9 also indicates three pillars and a suit of tools and artifacts, which enable us to infer activities in a detailed way. H10 does not present phases/activities, however, its strength involves its capacity for defining the delivered artifacts. Such artifacts suggest what phases/activities require to be executed as a process (such activities are considered in Table 2 as part of the

normalised activities of the ten GDPs). Some of the reviewed GDPs are based on a suite of methods and techniques that can be applied through the process. However, we did not include this characteristic in our characteristic list because such a characteristic is not directly related to the phases/activities.

In the case of characteristic ***PV2-2***, we identified the strengths and weakness regarding the integration of GDP into a SDLC. H1 to H7 do not propose an integrated process, specifically H1, H2, H5, H6 do not consider such an integration. H3 and H7 consider the possibility of non-separation. H4 is not integrated, but Nielsen [18] suggests that his proposal (i.e. H4) is easy to integrate with a SDLC in an intuitive way. On the other hand, H8, H9, H10 propose integrated processes.

A summary of the strengths and weakness of the ten GDPs involving a comparison with the GDP we propose in this work is presented in Section 5 (Figure 7).

3. A New GUI Development Process

Based on the analysis presented above we present an improved generic GDP. Our GDP provides phases and activities for each phase (see Table 3) and can represent a "natural evolution" from the analysed GDPs. Table 3 matches the proposed phases and activities of the GDP with the general macro-phases and phases of the SDLC.

The organization of Table 3 is the following, column 1 includes three macro-phases of SDLC (Definition, Development and Evolution), which are equivalent to the three phases of Nielsen's GDP [18] (see H4 in Table 1). Column 2 shows the following phases: Requirements, Design, Construction and Operation. Columns 3 and 5 of the Table 3 correspond to the phases and the activities of the GDP respectively. Finally, column 4 have the descriptions of each proposed phase of our GDP.

The resulting model structure of GDP (see Table 3) seems to be generic enough to easily "fit" into the SDLCs' Macro-Phases and Phases. Thus, Table 3 suggests that the GDP phases "Planning", "Analysis" and "Predesign" should be performed as part of the SDLC phase of "Requirements". Now, if we analyse the activities in the SDLC phase "Requirements" for a particular process model of SDLC, it is possible to detect duplicated activities in "Planning" and "Analysis" in GDP; therefore, it would be appropriate to identify and avoid such redundancy. A complete analysis of redundant activities between phases could represent an interesting avenue for future research works. To clarify the proposed generic GDP, we present the phases' activities and their descriptions in Table 4. Such descriptions are based on H9 that explains techniques to execute specific activities e.g. "Task Analysis", that is a common activity in HCI.

Table 3. Phases' descriptions and activities for the generic development process of the GUI –our GDP- matched with the macro-phases and phases of the SDLC

SDLC Macro-Phase	SDLC Phase	Proposed GDP Phases		Proposed GDP Activities
Definition	Requirements	Planning	In the phase Planning, the definition of the problem and system as well as definition of the plan is achieved.	Project Selection Marketing Activities Organize the Work
		Analysis	The phase Analysis focuses on understanding the users, their work concepts, and the needs of data and processes for their work. User scenarios can be created to structure interviews and use cases. Users help to construct their work domain models and information object models that are needed to perform their work. The analysis is human-centered and produces requirements of computational technology to increment and support human activities.	Analysis of Needs User Analysis Task Analysis Functional Analysis Requirements Specification
		Predesign	In the phase Predesign, an early design proposal highly-matched with the user expectations is defined. In this phase it is sketched the first low fidelity prototypes of the GUI.	Preliminary Specification of UI
Development	Design	Design	The phase Design begins with mapping the real world to the User Interface (UI). Computational specifications are assigned to GUI objects, including how they appear and behave on the screen. The general design, appearance, control and behavior of such objects are predictable and consistent with the user conceptual models. The original design will be refined several times based on of results of the evaluation.	Conceptual Design Formal Design Detailed Design
	Construction	Construc-tion	The phase Construction has the goal to establish evolvable implementations to a complete version of a system of high usability. Such a complete version is reached through prototype construction and empirical testing, ensuring it covers the final users' needs. This phase begins when a new version of the model of the GUI is rapidly constructed to show in a tangible way the designs concepts (prototyping). Early prototypes should include some interactivity for enabling users to assess the key issues. Prototypes help to understand the complete system and with reusable software components such prototypes will eventually evolve to the final delivered applications.	Prototyping (includes implementation of operational prototypes and proof of version) Deployment
Evolution	Operation	Evolution	Finally, the phase Evolution has the aim to collect final users' data about the current version of the software system. Such information is very important to improve the system in its subsequent versions or for employing such information in similar new systems. The software system is evaluated to determine to what extent the design supports the planed improvements made to the tasks. The evaluation can also be used to recommend how to improve the design. Learnability, throughput, and user satisfaction are the most common metrics for such an evaluation. This phase also considers assessable objectives of usability defined to provide the criteria for an acceptable design. When the criteria are satisfied the iterative design is complete.	Operation Evaluation Evolution

Table 4 describes the activities proposed in our GDP for each generic phase presented above in Table 3 and we claim can represent a "natural evolution" from the analysed GDPs.

Table 4. Phases and activities of the proposed GDP

	Activity	*Activity's Description*
Planning Phase	Project Selection.	This activity refers to the project formulation. In this activity it is defined the objective of the project and if the project will be subcontracted or not.
	Marketing Activities.	This activity refers to the execution of market activities, (i.e. carrying out an analysis of the software systems that are competing against the system under construction).
	Organize the Work.	This activity regards project planning involving the definition of activities, roles (including interdisciplinary roles like definers, designers, implementers, application writers, and manual writers that constitute large groups by themselves). Other activities involved include usage scenarios, preliminary meetings with prospective users, relevant prior experience, intense concentration on the design problem, and periods of non involvement with the design problem.
Analysis Phase	Needs Analysis.	In this activity, it is determined the type and objectives of the system, and the users.
	User Analysis.	In this stage, it is determined in a specific way the kind of user that will use the system, defining details such as the user's computer skills.
	Task Analysis.	In the task analysis, they are established the tasks that the user will perform with the system to reach their business objectives.
	Functional Analysis.	In this activity, they are defined the computing functions that are required to perform the tasks.
	Requirements Specification.	In this activity, formal specifications are described for the system implementation. Specifications like: Data dictionary, Entity-Relationship Diagram, etc.
Pre – design Phase	Preliminary Specification of UI.	In this activity, it is sketched the GUI design.
Design Phase	Conceptual Design.	The aim here is to design the system concept. This activity is repeated in an evolutive and incremental way.
	Formal Design.	This activity aims to construct the design models of the system with formal tools. This activity is repeated in an evolutive and incremental way.
	Detailed Design.	This activity aims to detail the formal design. This activity is repeated in an evolutive and incremental way.
Construction Phase	Prototyping	The goal of this activity is building any kind of prototype type that is necessary for the software development. For example, a prototype can be a high- or low fidelity prototype, an evolutive or disposable prototype, a global or local prototype, etc. There should, at least, be a global evolutive prototype that in the first versions could be of low fidelity and even disposable. Following this stage, a working prototype will be produced based on an iterative and incremental process. The prototype increments can be local prototypes that are integrated to the working prototype at a certain point of the development process. This activity includes testing each working prototype version. An evolutive prototype is a prototype that evolves to the final system by adding increments. A disposable prototype is a prototype that is no longer used to build the final system. A high-fidelity prototype is similar to the final system whereas a low-fidelity prototype involves initial prototype sketches.
	Deployment.	This activity refers to either release the evolving prototype or release the final system.
Evolution Phase	Operation, Evaluation & Evolution	At the end of each prototype increment, the working prototype is evaluated and the prototype evolution is planned.

3.1. Integration of the Generic GDP into a SDLC

To integrate our generic GDP with a SDLC, we propose to use the matching of proposed GDP phases and activities with the macro-phases and phases of a general SDLC. Such a matching is shown in Table 3. Therefore we have the activities -of the GDP and the process model of the SDLC-, that must be executed as part of the same phase of the

development process (although their order is not defined and it is possible to have some of the activities duplicated).

As an example we integrate the Spiral model [24] with the generic GDP by using the graphical definition of the Spiral model that presents four quadrants and several spiral loops containing the process activities. In order to simplify the location of phases/activities of the GDP over the Spiral, we first match the general SDLC (that is in column 2 of Table 3) over the quadrants of the Spiral, as shown in Figure 1.

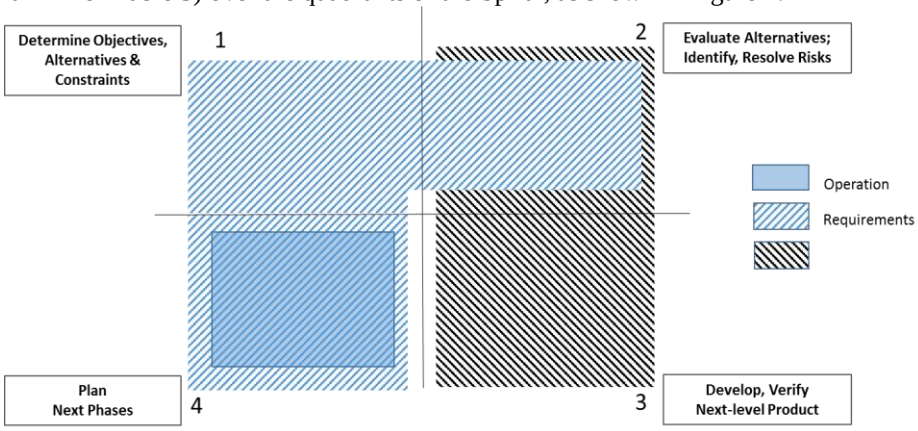

Fig. 1. General SDLC's phases matched with the Spiral process's quadrants

Such a match indicates in which quadrant of the Spiral model will be included the corresponding phases/activities of the GDP. Then over the quadrants we locate the generic phases of GDP according to Table 3, then:

- For the SDLC phase "Requirements" we have the generic GDP phases (1) "Planning", (2) "Analysis" and (3) "Predesign", which we allocate in the order (1), (2), (3) over the quadrants 1, 2, 3.
- For the SDLC phase "Design" and "Construction" we have the generic GDP phases (1) "Design" and (2) "Construction", which we allocate in the order (1), (2) over the quadrants 2, 3. This order is because in Spiral the activity "prototyping" is considered as part of design, but in GDP "prototyping" is part of construction.
- For the SDLC phase "Operation" we have the generic GDP phase (1) "Evolution", which we allocate over the quadrant 4.

According to the distribution above, the Figure 2 shows the location of the phases (planning, analysis, predesign, design, construction, evolution) of the generic GDP in the Spiral process model. Also, such a location is shown in Table 5:

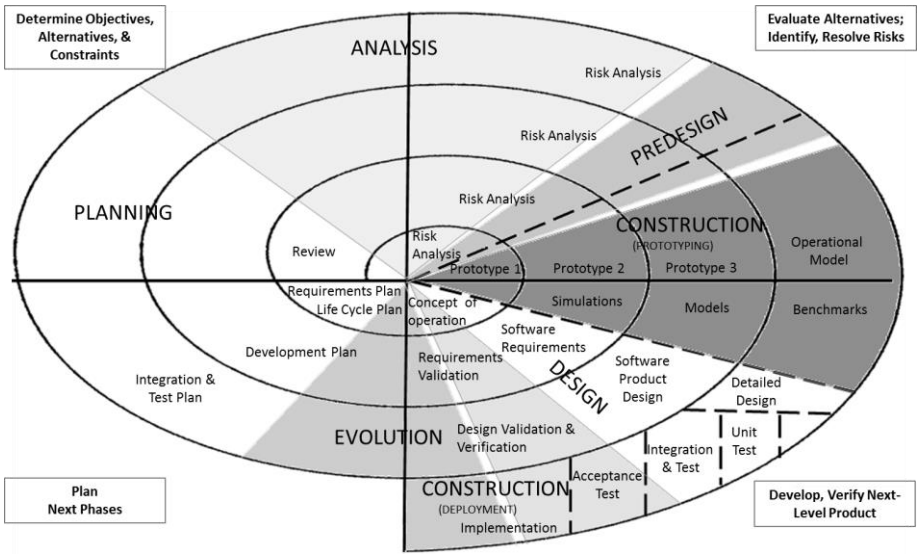

Fig. 2. Allocation of phases of the generic GDP in the Spiral Process

Table 5. Allocation of phases and activities of the GDP over the quadrants

Generic SDLC phases	Generic GDP phases	Quadrant 1. Determine Objectives, alternatives & Constraints	Quadrant 2. Evaluate alternatives, Identify, Resolve Risks	Quadrant 3. Develop, Verify Next-level Product	Quadrant 4. Plan Next Phases
Requirements	Planning	Project Selection, Marketing Activities,			GDP activities: Organize the Work
					SDLC activities: **Requirements Plan, Life Cycle Plan, Development Plan**
	Analysis	GDP activities: Needs Analysis, User Analysis, Task Analysis, Functional Analysis	GDP activities: Requirements Specification		
		SDLC Activities: **Review**	SDLC activities: **Risk Analysis,**		
	Predesign		GDP activities: Preliminary Specification of UI		
			SDLC activities: **Prototyping**		
Design	Design			GDP activities: Conceptual Design, Formal Design, Detailed Design	
				SDLC activities: **Simulations, Models, Benchmark, Concept of Operation, Software Requirements, Software Product Design, Detailed Design**	
Construction	Construc-tion			GDP activities: Prototyping, Deployment	
				SDLC activities: **Unit Test, Integration & Test, Requirements Validation, Design Validation & Verification, Implementation**	
Operation	Evolution				GDP activities: Operation, Evaluation & Evolution
					SDLC activities: **Integration & Testing Plan**

It can be seen in Table 5, what activities of the GDP should be included or be executed in each quadrant. For example, in the quadrant 1 we need to execute activities

of Planning and Analysis of the GDP. Then the activities that the quadrant need to include are: for Planning, "Project Selection", "Marketing Activities" and "Organise the Work"; and for Analysis, "Analysis of Needs", "User Analysis", "Task Analysis", "Functional Analysis" and "Requirements Specification".

4. Case Example

In this section we show how our model can be used independently of any SDLC by employing an illustrative example of the execution of the generic GDP. For this particular case, our generic GDP is focused on the process. That is, our model does not suggest when the guides and principles of HCI should be used, rather, the HCI indicates that the guides must to be used accordingly to the deliverable artifact to be constructed [23]. Following this, we apply a few set of guidelines to the use of the generic GDP.

First, to illustrate the execution of the activities of the Planning and Analysis phases of the generic GDP, we present parts of the documents of the Planning and Analysis phases in Table 6 and Table 7, respectively.

Table 6. Part of the planning-document

Activity	Part of the documentation generated as an example.
Project Selection.	**System description.** The system is a Web application of e-commerce that sales sport cycling articles. **Objective of the project development.** The objective of the project involves adding a mechanism to the system to ensure payment transactions are secure.
Marketing Activities.	Given that it is aimed to obtain a system that is competitive in the market, it is required to provide a mechanism that ensures the data of the users is safe.
Organize the work	Since this is a small-scale project only a few team members are required to carry out the changes: **Analyst-Programmer**. This member is required fulltime for two weeks. One week is needed for carrying out the system development. Another week is required for setting the system in operating mode, whereby any bugs found are immediately fixed. This member is responsible for designing any changes required by the user interface. However, this design is supervised by the GUI Designer. **Tester**. This member is required only partial-time for the first week. Note: Both, the ***Analyst-Programmer and Tester*** are responsible of planning the work for the life cycle.

Table 7. Part of the analysis-document

Activity	Part of the documentation generated as an example.
Needs and User Analysis.	In order to protect the user data and to make sure the online payments made on the e-commerce system are secure, two type of users are defined as follow: **Customer.** Person making use of e-commerce services through the Internet whereby she/he is able to select offered products and pay for them. **User.** Person making use of e-commerce services through the Internet whereby she/he is able to view the products offered, select products, and put products in a shopping-cart. This user is converted into a "Customer" when it has at least a product in the shopping-cart and intends to pay for such products.
Task and Functional Analysis.	General Use Cases Descriptions.- (Figure 3 shows the Context Use Case Diagram.) **Use Case: The user** selects products and puts them in the shopping-cart. *General Use-Case description:* The user selects the products from the e-commerce site and puts them in the shopping cart. **Use Case:** Sale payment. *General Use-Case description:* The "Customer" captures the data for paying the products that the user has selected and puts in the shopping-cart. **Use Case:** "Secure mode" for payments: Ensuring and verifying the "secure mode" of the system to ensure secure payment transaction over the Internet. *General Use-Case description:* This case aims to ensure a "secure mode" of the system to receive payments protecting the user's data. This use case will be executed when one person (customer) of the e-commerce system intends to pay for several products previously selected. Only when the "secure mode" is enabled, the system verifies and protects all data captured for the payments. The user must indicate if the system must enable or disable the "secure mode" by the use cases "enable secure mode" and "disable secure mode". The system maintains visual indicators about the following states: "secure mode" and "unsecure mode". When the capture mode is in "secure mode", the system indicates whether data problems are found by sending "error-messages" or "warning-messages". **Use Case:** Enable "secure mode" *General Use-Case description:* This use case enables a "secure mode" state for capturing payments, and sets on the indicator as a "secure mode" state. **Use Case:** Disable "secure mode" *General Use-Case description:* This use case disables a "secure mode" state for capturing payments, and sets on the indicator for the capture mode as non "secure mode". **Use Case:** Information about "secure mode" *General Use-Case description:* This use case shows information about the

	capture modes when the customer enables or disables the "secure mode". This use case is also about how the customer can know when the "secure mode" is enabled or disabled. Flow Diagrams.- Figure 4 shows the Data Flow Diagram for the process of "Sale Payment".
Requirements Specification.	**Non Functional System Requirements:** It is required a tool for network monitoring. **Functional System Requirements:** This case begins when the user (customer) intends to capture her/his confidential payment data. At the moment, the system shows "warning" and "error" messages, and indicates when the system is in the "secure mode" for paying.

Now we exemplify the phases of Predesign and Design in Table 8 and Table 9 respectively. As follows.

Table 8. Part of the predesign-document

Activity	Part of the documentation generated as an example.
Preliminary Specification of UI	It is important to mention that for this example we considered the design criteria for secure software interaction proposed by [12], which were applied through the GDP specifically during the *Predesign Phase*. The use of these criteria is merely illustrative and could be replaced by any other design specification. Particularly, we chosen these criteria because they joint the basics for usable secure software development and the traditional Nielsen's usability requirements: *Visibility of system status, Aesthetic and minimalist design, Satisfaction, Convey feature, Learnability, Trust* [12], [18], which contribute in achieving good designs. In this way good designs are those that fulfill the specifications that determines the shape and strengths for a product or service. These aspects are referred into Conceptual Design, which - in terms of software- normally consider elements such as User Interfaces Design, navigation, Security, among others [17], [5].

Table 9. Part of the design-document

Activity	Part of the documentation generated as an example.
Conceptual Design.	From applying some need findings techniques; such as observation and interview [23], [26]; we could obtain feedback that may complement very well to results from Predesign phase. This information was then distilled throughout an interaction sketch/ paper prototype (see Figure 5).
Formal Design.	We matched the feedback provided by the Conceptual Design activity to the specific knowledge provided by the set of design patterns presented in [17]. The use of design patterns (and/or other similar approaches) could provide formality to the design process. The results from "Formal Design" activity consist of a digital mock up and its specification. Here is presented a fragment of the specification for elements to convey the security level of the system.

	Specification for screen (that evolves to Figure 6A): • *Problem*: How can the security features of the web service be clearly stated to the users? • *Solution*: The users will be alerted about the protection of the system by using an image of traffic lights and showing the message "The Security Module is ACTIVE". A green colour in the screen frame and the green light turned-on in the traffic lights are used to indicate the users that the system is protected. The text "The Secure Transaction is ACTIVE" is always visible. In the same way, a message is presented in a dialogue box that also includes the option to disable the security module or to continue using it giving the user more control over the system. Specification for a second screen (that evolves to Figure 6B): • *Problem*: How to present a clear visibility of the system status? • *Solution*: (it is not included for the sake of brevity)
Detailed Design	The digital mock up produced through the "Formal Design" activity, complemented with additional feedback from users would provide a reliable starting point for a more detailed design which could be materialized using a semi-functional digital mock up closer to the final construction version (see Figure 6A y 6B).

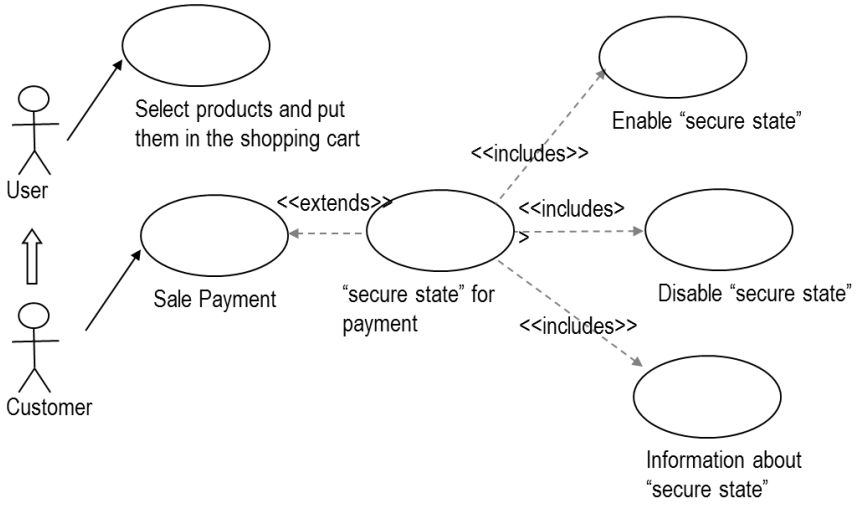

Fig. 3. Contextual Use Case Diagram

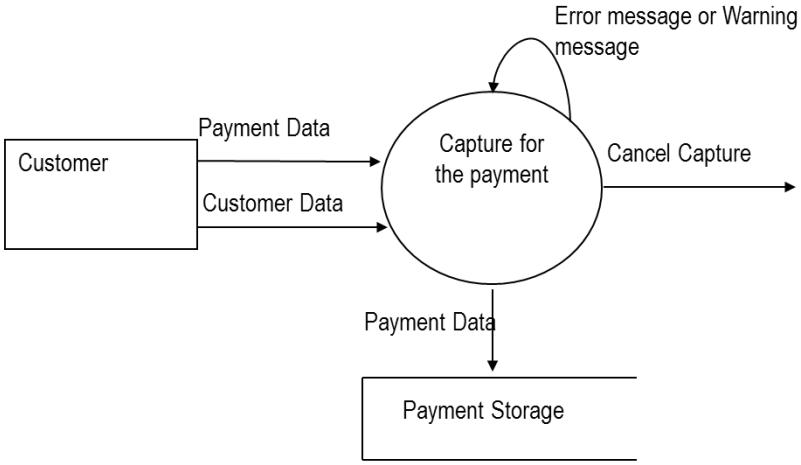

Fig. 4. Data Flow Diagram for the "Sale Payment" process

Finally, Table 10 shows an example of the execution of the activities for the Construction and Evolution phases.

Table 10. Part of the construction-document

Activity	Part of the documentation generated as an example.
Construction.	From the specifications provided by the design phase, we constructed the user interface for the specific system incorporating all the elements in a prototype. As part of the "Construction" stage the prototype (that is by space is not presented here), generated and approved by the development team, is transformed into a functional Interface, which is then launched as an evolving prototype and eventually the final system.
Evolution.	The final version of the interface could be evaluated now, from both an operational and usable points of view. The evaluation performed could lead to a reverse engineering process, even being able to incorporate new design requirements encouraging enhancements of the entire user interface. This evolutionary process could be easily incorporated into basic SDLCs through the phases of our model.

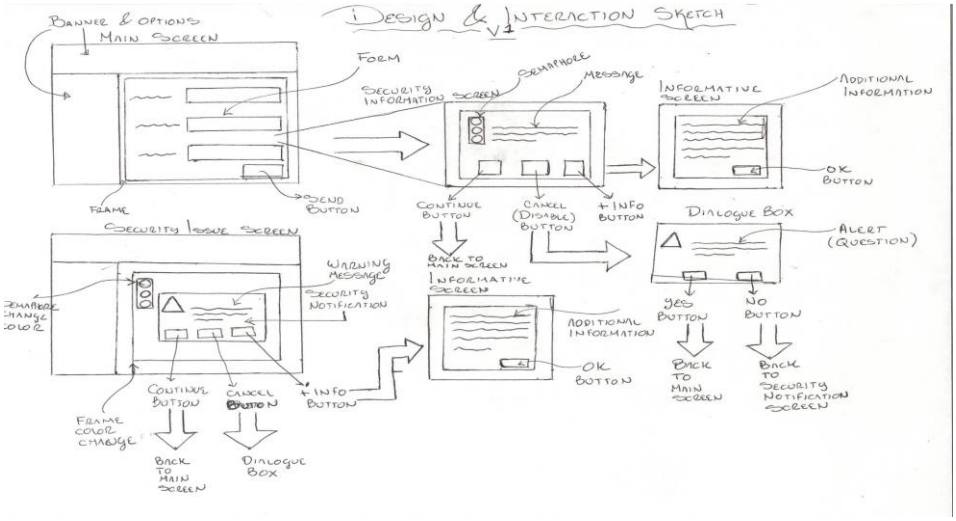

Fig. 5. System Design and Interaction Sketch

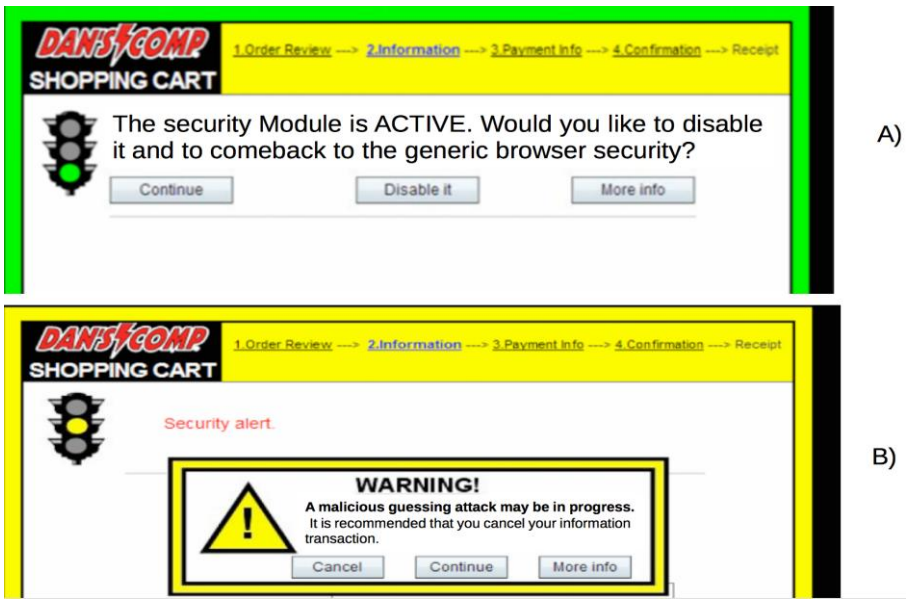

Fig. 6. A) Conceptual design for Screen 1 B) Conceptual design for Screen 2 B)(courtesy of www.danscomp.com)

5. Evaluation

From a methodological point of view, it is necessary to describe the process of how we can develop software products avoiding "assumptions" as much as possible in order to succeed in the development of a software product independently of the developers' expertise. As in other areas of science, software engineering knowledge grows incrementally; that is, frequently, new proposals represent evolutions of past contributions. On this basis, we analysed the strengths and weaknesses of ten process-oriented Graphic User Interface Development Process (GDP) models. This study allowed us to identify some gaps. The general GDP we have proposed addresses some of these gaps.

The literature review revealed several opportunities for improvement, hence, we have merged some virtues of the best GDPs analysed together with additional features obtained from our experience in developing and designing systems as well as analysing development processes; resulting in a new, basic but more robust GDP. Our main contributions include the following:

- Identification of GDPs activities: The definition/description of GDP activities is essential in order to obtain a complete and understandable explanation of a development process, and must be consistent with the stages to create a graphic user interface.
- Integration of the GDP into the SDLC: It is common to talk about GDPs and SDLCs in a separate way, since Human Computer Interaction (HCI) commonly studies the characteristics and behaviors of GDPs, and Software Engineering (SE) deals with the study of SDLCs. We believe as some other authors do [21], [22] that the Software System development process should not be separated from the development process of its own GUI.

We evaluated the different GDPs reviewed, including ours, with *Fully met*, *Partially met* or *Not met*, depending on whether the GDPs met each one of the four characteristics (PV2-1, PV2-2, PV2-3, PV2-4) defined in Section 2. We performed the evaluation in an independent way of the SDLC that could be considered for a possible the integration.

Figure 7 shows the evaluation represented as four bars (one bar for each analysed characteristic) for each GDP: the horizontal axis represents each GDP (i.e. H1..H10 and our proposed GDP) and the vertical axis represents the three levels of meeting a characteristic (i.e. Fully met, Partially met, Not met).

For the evaluation analysis we defined two objectives. We defined objective one (O1) as a GDP's well-defined modelling process. O1 is achieved by satisfying characteristics PV2-1 and PV2-3. We also defined objective 2 (O2) as the integration of a GDP within a SDLC. O2 is achieved by satisfying characteristic PV2-2. Bars in Figure 7 were rated with values 0.33, 0.66, or 1 if a given characteristic is not met, partially met, and fully met, respectively. The coverage of O1 obtained by a GDP was calculated by obtaining the level of support given to both PV2-1 and PV2-3, i.e. the average of both values. For example, in the case of H5, we have that the value for PV2-1 = 0.33 and for PV2-3 = 0.33. Hence, we have that H5's coverage of O1 is given by (PV2-1 + PV2-3)/2 = (0.33 + 0.33)/2 = 0.33. The coverage of O2 achieved by a GDP is directly obtained from the level of support given by characteristic PV2-2. The values for such objectives are presented in Table 11.

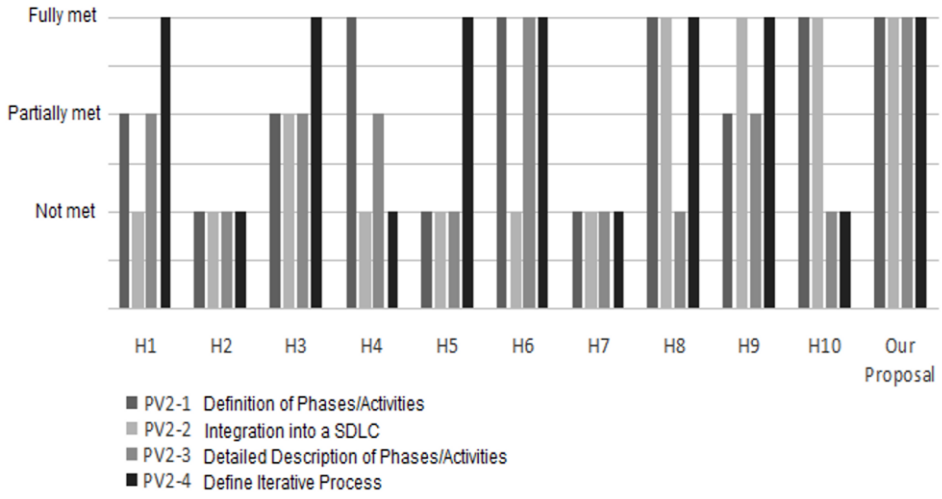

Fig. 7. Analysis of the ten GDPs for detecting weakness

Table 11. GDPs' coverage of objective O1 (well-defined process) and objective O2 (integrated into SDLC)

	O1 - Well-defined process (referred to phases and activities)			O2 – Integration of GDP into SDLC	
	PV2-1	PV2-3	Coverage for O1	PV2-2	Coverage for O2
H1	0.66	0.66	0.66	0.33	0.33
H2	0.33	0.33	0.33	0.33	0.33
H3	0.66	0.66	0.66	0.66	0.66
H4	1.00	0.66	0.83	0.33	0.33
H5	0.33	0.33	0.33	0.33	0.33
H6	1.00	1.00	1.00	0.33	0.33
H7	0.33	0.33	0.33	0.33	0.33
H8	1.00	0.33	0.66	1.00	1.00
H9	0.66	0.66	0.66	1.00	1.00
H10	1.00	0.33	0.66	1.00	1.00
Our GDP	1.00	1.00	1.00	1.00	1.00

Columns fourth and sixth of Table 11 show the values of O1 and O2, respectively. We can observe that most of the previous proposals have focused on identifying the phases or activities, but do not provide details of them. In this paper, we define phase details as both the phase description and the definition of activities whereas activity definitions refer to the description of activities. The last three proposals i.e. H8, H9, and H10 have evolved towards the integration with a particular SDLC, i.e. the waterfall model [24]. However, none of them explains how such an integration can be achieved with an arbitrary SDLC. In contrast, although our approach is not integrated with any

particular SDLC, our GDP is more generic as it defines the integration process with any SDLC.

Our proposed GDP is the first effort to fully achieve objectives O1 and O2. The case example presents a feasible use of this generic GDP by employing tools and terms previously defined in Software Engineering (SE), (e.g. the "use case" of UML, or the term "Non Functional System Requirements" [24]) or techniques, methods and usability guidelines previously defined in HCI (e.g. the use of "sketching" [9], or the application of "security design criteria" [12]).

5.1. Our Proposed GDP vs the ten reviewed GDPs

Before anything we remark that the activities of our proposed GDP are taken of the previous H1..10. Bearing in mind, we argue in favor of our proposal as a natural evolution, not as replacement of the previous GDPs.

Our proposed GDP has the following advantages over the analysed GDPs. Firstly, other related and emergent solutions (such as [6]) are presented as a set of deliverables or product models of the process instead of a process itself. In contrast, our proposal is focused on defining the phases and activities of the GUI development process as part of a well-defined process. Secondly, despite the proposal of Nielsen (H4) includes processes, Nielsen's work defines the activities only as elements to be covered in the process and no distinction is provided between methods and processes. Also, from the point of view of a well-defined process, it is necessary the specific identification of activities. Finally, regarding characteristic 2 (integration into a SDLC) of Figure 7, we can see that the proposals H8, H9 and H10 explicitly indicate the integration of a GDP into a specific Software-System Development Life Cycle (SDLC), i.e. the waterfall model [24]. The proposal H8 provides a way to integrate a GDP with a specific SDLC. H9 presents the integration as a black box, and H10 integrates both processes, GDP and SDLC as a single process, but not in an explicit way for its execution. In contrast, in our proposal, the integration is not obtained in an ad hoc manner, rather, the integration can be systematically achieved into any SDLC.

Considering the case example, we provide below an analysis that compares our GDP with other GDPs.

- The planning phase of our GDP is exemplified in Table 6 that has several planning activities that should be an overview of the system and its objectives, an overview about the marketing possibilities for the system and how the work can be organized to develop the system, including roles and their great responsibilities.

Following, we compare the planning phase of H8, H9 and H10 with our GDP:

H8 and ***H9*** do not considerate the planning proposed in the activities exemplified in Table 6, they go directly to the systems requirements that for our GDP begins with the analysis phase.

H10 considers the planning phase as a single large activity called *Project Selection and Planning* that includes only the activities of Project Selection and Project Planning. This last activity is equivalent to the Work organization activity of our GDP.

- The analysis phase of our GDP, exemplified in Table 7, gives the first specification for the requirements of the system.

Following, we compare the analysis phase of H8, H9 and H10 with our GDP:

Although ***H8, H9*** and ***H10*** have an equivalent final activity *–Requirements Specification–* for the analysis phase, only ***H10*** have five activities in the analysis phase which are equivalent to the activities defined by our GDP (i.e. *Analysis of Needs*, *User Analysis*, *Task Analysis*, *Functional Analysis*, *Requirements Specification*) in such a phase. H8 has only two activities (*Task Analysis and Requirements Specification*).

- Our proposal has four activities in the predesign and design phases: *preliminary specification of UI* (that helps to a better understand of the requirements), *conceptual design*, *formal design* and *detailed design*. Four activities that promotes different levels of design and gives a small process to follow. In contrast, F8 defines only two types of design. H9 and H10 define only a large activity for design phase.
- The activity of prototyping is common in the proposals H8, H9 since it is the base of a good GUI design. They consider the prototype, only for evaluation, i.e. not for implementation. But, H10 does not consider prototyping.

Finally, we are proposing the way to achieve integration. We claim that others GPDs can use our way to integrate with an SDLC. The activities of others GDPs are normalized in Table 1, therefore, we can match them with the macro-phases and phases of SDLC and integrate all of them with any SDLC. However, even when we use this way for integration, most of the GDPs are not enough for a good integration because they are not sufficiently detailed. For example we can easily integrate H10 with our integration approach, but this is not the case of H4, which is not sufficiently detailed.

5.2. Limitations

It is important to highlight that this GDP is proposed from the point of view of a methodological processes, without proposing HCI principles and guidelines of design. However, we claim that the principles and guidelines defined by the HCI community can be used with the proposed generic GDP. Then this work presents the basis for obtaining a well-defined GDP. The GDP presented is limited to the definition of a minimum process that includes the phases, activities, their descriptions, and the iterative mode of the process. To achieve a complete and well-defined GDP from a methodological view point, it is necessary to define the deliverable artifacts, roles and the applicable tools (methods, techniques, and technology) that include a plausible integration of principles and guidelines of design and analysis of the process-oriented GDPs with respect to their deliverable artifacts, roles and tools.

Finally, another limitation of our work is that the possible strengths of some good proposals of model-oriented GDPs are not considered here since our study is focused on process-oriented GDPs.

6. Conclusion

In order to apply the HCI experience in a development process it is commonly necessary to be an expert in both knowledge areas, HCI and SDLC. To address this issue, research in the HCI field has proposed approaches that intend to include the SDLC in their methodological proposals. However, some of these proposals only present the models (of artifacts) to be constructed in each phase of the SDLC and do not define the activities implied to construct the HCI models [6]. On the other hand, in the SE field the activity of screen design is generally defined as a single activity and regarded as a black box in the system design. In this paper, we have proposed a generic GDP that enables the integration of such GDP in an arbitrary SDLC. Our work has three main contributions: (1) identifying the activities that can be generic for a GDP; (2) defining a generic GDP including the experience of several methodological approaches in HCI; and (3) an approach towards the integration of GDP and SLDC.

As a future work, we will look into defining a complete well-defined process model for GUI development, which involves complementing the GDP with the definition of the products or models of products of each activity, the roles for the process and the a suit of suggested tools to use for the process execution.

References

1. Abd Majid, R., Md Noor, N. L., Wan Adnan, W. A., and Mansor, S.: A Survey on HCI in the Software Development Life Cycle: from Practitioner's Perspective. ACM, 978-1-60558-710-3/09/11, 1–4. (2009)
2. Bailey, G.: Iterative Methodology and Designer Training in Human-Computer Interface Design. ACM INTERCHI'93, 198-205. (1993)
3. Blomkvist, S.: User-Centered Design and Agile Development of IT Systems. IT Licentiate theses, Department of Information Technology, Uppsala University (2006).
4. Butler, A.: Usability Engineering Turns 10. ACM Interactions, 59-75. (1996)
5. Chemuturi, M.: Master Software quality assurance: Best practices, tools and techniques for software developers, J. Ross Publishing. (2010)
6. Giraldo, W.J., Molina, A.I., Collazos, C.A., Ortega, M., and Redondo, M.A.: A Model Based Approach for GUI Development in Groupware Systems. In: Briggs, R.O., Antunes, P., de Vreede, G., and Read, A.S. (ed.): Groupware: Design, Implementation, and Use. 14th International Workshop, CRIWG 2008, Omaha, NE, USA, September 14-18. (2008), Revised Selected Papers, Lecture Notes in Computer Science Volume 5411, Springer, 324-339. (2008)
7. Gould, J., Lewis, C.: Designing for Usability: Key Principles and What Designers Think. Communications of the ACM, Vol. 28 No. 3, 300-311. (1985)
8. Göransson, B., Gulliksen, J., and Boivie, I.: The Usability Design Process – Integrating User-centered Systems Design in the Software Development Process. John Wiley & Sons, Ltd., Software Process Improvement and Practice, 111–131. (2003)
9. Granollers, T., Lorés, J. & Perdrix, F.: Usability Engineering Process Model. Integration with Software Engineering. In Proceedings of HCI International 2003, Crete, Greece, June 22-27-2003, Lawrence Erlbaum Associates, New Jersey, USA, 1-6. (2003)
10. Gulliksen, J., Göransson, B., Boivie, I., Blomkvist, S., Persson, J., and Cajander, A.: Key Principles for User-Centred Systems Design. Taylor & Francis Ltd., Behaviour & Information Technology, Vol. 22, NO. 6, 397–409. (2003)

11. Hartson, H. R., and Hix, D.: Human-Computer Interface Development: Concepts and Systems for its Management. ACM Computing Surveys, Vol. 21 No. 1, 5-92. (1989)
12. Johnston, J., Eloff., J., and Labuschagne L.: Security and Human Computer Interfaces. Computers & Security, 675-684. (2008)
13. Losada, B., Urretavizcaya, M., and Fernández-Castro, I.: The InterMod Methodology: An Interface Engineering Process Linked with Software Engineering Stages. In: Macías, J.A., Granollers Saltiveri, A., and Latorre, P.M. (ed.): New Trends on Human-Computer Interaction. Research, Development, New Tools and Methods, Springer-Verlag London Limited, 53-63. (2009)
14. Lowdermilk, T.: User Centered Design, O'Reilly Media. (2013)
15. Lowgren, J., and Stolterman, E.: Design Methodology and Design Practice. ACM Interactions, 13-20. (1999)
16. Mora, M.: The Conceptual Research Approach. Universidad Autónoma de Aguascalientes, (2005).
17. Muñoz, J., Mendoza, R., Vargas Martin, M., Vanderdonckt, J., Álvarez, F., and González Calleros, J.: A method to design information security feedback Using patterns and HCI-security criteria. In Proceedings of the 7th International Conference on Computer-Aided Design of User Interfaces CADUI, Lecture Notes in Computer Science, Springer, Albacete, Spain, (2008)
18. Nielsen, J.: The Usability Engineering Life Cycle. IEEE, 11-22. (1992)
19. Oktaba, H., and Ibargüengoitia, G.: Software Process Modeled with Objects: Static View. Computación y Sistemas. Vol. 1 No. 4 CIC-IPN ISSN 1405-5546, 228-238. (1998)
20. Rodríguez L. C., Mora M., and Alvarez F. J.: Process Models of SDLCs: Comparison and Evolution. In: Mahbubur Rahman S., Nessa Syed S. (Eds.), Research con Modern Systems Analysis and Design Technologies and Applications. IGI-Global, pp. 76-89. (2008)
21. Rodríguez-Martínez, L. C., Mora, M., Muñoz, J., and Mendoza, R.: A Descriptive-Comparative Study of Activities of GUI Development Processes and their Evolution. In Proceedings of CONISOFT, 17-24. (2012)
22. Rosson, M., Maass, S., and Kellog, W.: Designing for Designers: Analysis of Design Practice in the Real World. ACM CHI+GI 87, 137-142. (1987)
23. Shneiderman, B., and Plaisant, C.: Designing the User Interface: Strategies for effective Human-Computer Interaction, Addison Wesley. (2010)
24. Sommerville, I.: Software Engineering, Addison-Wesley. (2005)
25. Wallach, D., and Scholz, S.C.: User-Centered Design: Why and How to Put Users First in Software Development, Springer-Verlag Berlin Heidelberg., Software for People, Management for Professionals, 11-38. (2012)
26. Dix, A.: Human Computer Interaction, 3rd Edition, Prentice Hall. (2004)

Laura C. Rodriguez-Martinez is a full Professor at the Systems and Computing Department, Instituto Tecnologico de Aguasaclientes, Mexico. She holds a PhD in Computer Science at Universidad Autonoma de Aguascalientes, Mexico in 2009. Her research interests include Software Systems Development Processes, Service-Oriented Software Engineering and Graphical-User Interfaces Development Processes.

Hector Duran-Limon is currently a full Professor at the Information Systems Department, University of Guadalajara, Mexico. He completed a PhD at Lancaster University, England in 2002. His research interests include Cloud Computing and High Performance Computing (HPC). He is also interested in Software Architectures, Software Product Lines and Component-based Development.

Dr. Ricardo Mendoza-González is a researcher professor at Universidad Politecnica de Aguascalientes and the Instituto Tecnologico de Aguascalientes. He holds a PhD in Computer Science and particularly addresses his research works to Human-Computer Interaction and UX topics. He has over forty scientific publications in Journals, book-chapters, and conference proceedings.

Dr. Jaime Muñoz Arteaga is a research professor at Universidad Autonoma de Aguascalientes, Mexico. He got his PhD in Computer Science at Université Toulouse III, in France. His research topics are in Human-Computer Interaction, E-learning and Web Engineering. The Dr. Muñoz has published two books about Software Engineering, one book about Human-Computer Interaction and two books about Learning Object Technology.

Received:March 01, 2014; Accepted: January 07, 2015