A Panorama of Software Architectures in Game Development
Leonardo Bitencourt Morelli
Elisa Yumi Nakagawa
Dept. of Computer Systems
Institute of Mathematics and Computational Science - ICMC
University of São Paulo - USP
morelli,elisa@icmc.usp.br

Abstract—As video games evolve into richer and more sophisticated products, the software driving those games become more complex. One of the research areas offered by Software Engineering to cope with this complexity, while reducing risks and improving software quality, is Software Architecture. The purpose of this paper is to present an overview of possibly all work having investigated, established and used software architectures for the development of video games. For this, a Systematic Mapping was conducted. The achieved results show an increasing, however still mild, interest in the exploration of software architectures for the development of video games, and lays out lines of research that can be explored.

Keywords- Software Architecture; Game Development; Video Game; Computer Game; Systematic Mapping

I. INTRODUCTION

Developing video games has evolved from a one-man job into highly complex projects, executed by large groups of highly specialized professionals [4] over the period of years. This reflects the fact that the video games industry is the fastest growing one in the entertainment market [17], with titles (i.e., individual games) selling millions of copies in the week immediately following each release date [5]. Both the growing complexity and the diversity of target platforms require efforts from the development team to improve software quality and reduce project risks, as well as minimize the cost of a product and make sure it meets today’s market standards.

One of the Software Engineering disciplines that can address these efforts is Software Architecture [18], commonly associated with software quality and maintainability. From the first work of Kruchten on iterative software development with a focus on software architecture [10], a number of works has recognized the value of considering software architectures explicitly in system development processes [2], [19]. As already stated by Shaw and Clements [15], software architectures have attained the status of truly successful disciplines. Software architectures play a major role in determining system quality — performance, maintainability, and reliability, for instance — since they form the backbone of any successful software-intensive system [18].

According to Garlan [8] and the SEI [16], a Software Architecture consists of a component structure, the relations between components, and the rules governing the design and evolution of software systems. A well-defined software architecture can improve system reliability, enable the integration of third-party libraries and COTS(Commercial, Off-The-Shelf) components, and promote component reuse, reducing production costs between projects. Because of this, numerous application domains have been the subject of software architecture studies, and so should video games. It is, therefore, interesting to search for previous works associating software architectures with video game development.

Evidence-Based Software Engineering (EBSE), which is inspired in the medical area and has contributed considerably to software engineering practices, gives us the Systematic Mapping technique. It offers a systematic method for mapping a given research topic in order to obtain a comprehensive overview of the area[13].

This paper’s main objective is, through the conduction of a Systematic Mapping Study, to identify possibly all research works investigating the use of software architecture in video game development. While few such publications currently exist, results point out that interest in the area is growing and reveal interesting specific topics that haven’t been explored.

This paper is organized as follows. In Section II we present a brief overview on the Systematic Mapping technique. In Section III we present the conducted Systematic Mapping. In Section IV we discuss about achieved results. Finally, in Section V we summarize our contributions and discuss perspectives for further work.

II. A BRIEF OVERVIEW OF SYSTEMATIC MAPPING

When a research field reaches its maturity, there is almost always a noticeable increase in the number of reports and results which are made available. At some point in the study of new areas, researchers usually conduct bibliographical reviews (almost always informal ones) to identify publications corresponding to their specific subjects. These reviews are not, however, done systematically, nor is any support given to prevent bias from occurring during the selection of publications set to be analyzed. It is important to have mechanisms to summarize and provide overviews of an area or topic of interest [13].
For this, EBSE has proposed the use of the Systematic Mapping technique [3], [13]. In this context, any individual evidence (e.g., a case study or an experimental study divulged in a publication or paper) which contributes to a systematic mapping is called a primary study, while the result of a systematic mapping is a secondary study. Systematic mapping aims at providing an overview of a research field, assessing the quantity and types of primary studies existing in the area of interest [13]. In short, systematic mapping is carried out by planning, conduction of search and screening of primary studies by use of inclusion and exclusion criteria [3]. A systematic mapping also includes data extraction and analysis through the identification of categories and classification of primary studies within these categories. As a result, this technique provides maps (i.e., tables and charts) containing condensed information on the area of interest.

III. Conducted Systematic Mapping

This systematic mapping aims at identifying studies describing the use of software architectures in the development of video games, and it was conducted between May and June, 2010, using the process shown in Fig. 1. It is, in short, composed by four stages: (i) systematic mapping planning; (ii) conduction of the search; (iii) selection of the primary studies; and (iv) analysis, classification and mapping. These steps are explained in further detail in the next sections, which present the conducted systematic mapping and explain its execution.

A. Systematic Mapping Planning

This plan consists of formulating the research questions, selecting the sources of primary studies and establishing the inclusion and exclusion criteria.

1) Research Questions (RQ): These questions are structured according to the objective of the systematic mapping. In this case, their purpose is the identification of a panorama featuring both software architectures and video game development. The research questions for our systematic mapping are:

RQ1: Which software architecture topics have been studied for the development of video games?
RQ2: Who are authors researching the use of software architectures for the development of video games?

RQ3: Which software architecture topics for video game development have been given more attention?
RQ4: Which video game subsystems have been targeted in software architecture research?

2) Selection of Sources: As the sources of primary studies, the databases indexing the most important publications in software engineering were selected: IEEE Xplore1, ACM Digital Library2 Springer Link3, Scopus4, Science Direct5 and ISI Web of Knowledge 6. Only papers written in English were considered in the systematic mapping, since English is the most widely adopted language in the publication of scientific papers.

3) Establishment of Selection Criteria: The definition of Inclusion Criteria (IC) and Exclusion Criteria (EC) is another important element in the planning of systematic mapping. These criteria make it possible to include primary studies that are relevant for the research questions and to exclude studies that do not answer them. The inclusion criteria in our systematic mapping are:

IC1: The study describes the investigation of a software architecture for video games;
IC2: The study describes the investigation of a reference architecture for video games;
IC3: The study describes the investigation of a framework for video games;

The exclusion criteria established are:
EC1: The study is not in English;
EC2: The study does not have an abstract;
EC3: The study is in abstract-only format.
EC4: The study does not describe the investigation of a software architecture.
EC5: The study does not describe an investigation about video game development.
EC6: The study is a copy or older version of another considered study.

B. Conduction of the Search

At this stage, the search for primary studies is conducted according to the previously established plan, i.e., by searching for all primary studies matching a search string in the databases selected as sources. This can be done automatically if the sources provide an efficient search engine. For this, useful keywords are selected and search strings are created.

1) Keywords: To keep studies within the systematic mapping’s scope, the keywords must be simple and well-chosen. They must be sufficiently simple to produce an adequate number of results, but also rigorous enough to avoid the inclusion of undesired studies. The keywords

1http://ieeexplore.ieee.org/
2http://portal.acm.org/
3http://www.springerlink.com/
4http://www.scopus.com/
5http://www.sciencedirect.com/
6http://www.isiknowledge.com/
chosen for this systematic mapping, separated by area, are shown in Table I.

### TABLE I
**Key Words Separated by Area**

<table>
<thead>
<tr>
<th>Area</th>
<th>Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Architecture</td>
<td>“software architecture”</td>
</tr>
<tr>
<td>Video Games</td>
<td>“video game”, “computer game”</td>
</tr>
</tbody>
</table>

2) **Search Strings**: Using the keywords, the search string is built with AND/OR operators in a way that it represents the correct search parameters. The search string used in our systematic mapping was:

(`software architecture` OR “software architectures”) AND
(`video game” OR “video games”) OR (“computer game” OR “computer games”)

This search string was accepted by all selected databases, so no custom strings were necessary. With the Springer Link search engine, application of filters after the search was required. The following two filters were used: (i) in “Subject” the filter “Computer Science”; and in “Subject” the filter “Software Engineering”. With the ISI Web of Knowledge database, the string was used in “topic or title”.

This stage resulted in a total of 276 primary studies: 70 from ACM digital library, 12 from IEEE Xplore, 77 from Springer Link, 95 from Science Direct, 6 from Scopus and 16 from ISI Web of Knowledge.

### C. Selection of Primary Studies

At this stage, the inclusion and exclusion criteria were employed to select the relevant primary studies.

Table II summarizes the number of primary studies obtained after applying the inclusion and exclusion criteria. To give the databases a fair comparison, this table does not take EC6 (which excludes duplicated studies) into account. From a total of 276 primary studies previously identified, 33 studies (i.e. 12 %) were considered relevant and thus selected. ISI Web of Knowledge, Scopus and IEEE Xplore were very useful sources for this study: They had high inclusion rates, even though the number of returned studies was low. Ten of the studies returned by Springer Link were included and, despite the low inclusion rate, it should also be considered relevant. Most of the excluded studies didn’t cover software architectures or video games at all.

### TABLE II
**Partial and Total Amounts of Primary Studies Included and Excluded, Except by EC6**

<table>
<thead>
<tr>
<th></th>
<th>incl</th>
<th>excl</th>
<th>total</th>
<th>% incl</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE Xplore</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>33.3%</td>
</tr>
<tr>
<td>ACM Digital Library</td>
<td>3</td>
<td>67</td>
<td>70</td>
<td>4.3%</td>
</tr>
<tr>
<td>Springer Link</td>
<td>10</td>
<td>67</td>
<td>77</td>
<td>13%</td>
</tr>
<tr>
<td>Science Direct</td>
<td>4</td>
<td>91</td>
<td>95</td>
<td>4.2%</td>
</tr>
<tr>
<td>Scopus</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>50%</td>
</tr>
<tr>
<td>ISI Web of Knowledge</td>
<td>9</td>
<td>7</td>
<td>16</td>
<td>56.3%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>33</td>
<td>243</td>
<td>276</td>
<td>12%</td>
</tr>
</tbody>
</table>

For the remainder of the Systematic Mapping, EC6 is also applied, excluding duplicated results from the relevant primary studies. If there were two revisions of the same primary study, the oldest was discarded. The updated relevant primary studies are shown in Table III.

### D. Analysis, Classification and Mapping

While the previous section presented a quantitative analysis of the systematic mapping, this section categorizes the primary studies. For this, two tasks were conducted: (i) search by keywords; and (ii) grouping of the primary studies into categories. To start with, primary studies had their titles, keywords and abstracts carefully read. Following this procedure, terms that seemed to be more relevant were selected as keywords, and these keywords are divided into categories. For this systematic mapping, there was an interest in identifying relationships between software architecture and the development of video games. Thus, four categories, each with a group of keywords, were created:

**Category 1: Video game subsystems**: This category shows which video game subsystems software architectures have been proposed for. Fig. 2 illustrates these subsystems and their relationships in a common game system. The subsystems considered here are the result of empirical observation, both by the authors and by the game development community [12]. The keywords for this category are:

- **graphics**: Studies that describe the subsystem in charge of graphics rendering and display;
- **audio**: Studies that describe the subsystem in charge of audio, e.g., music and sound effects;
- **input**: Studies that describe the subsystem in charge of capturing user input from devices, e.g., controllers, mouse and camera;
- **network**: Studies that describe the subsystem in charge of network communications;
- **physics**: Studies that describe the subsystem in charge of physics simulations. This may include the use of a dedicated physics processor or GPU (Graphics Processing Unit) processing;
- **AI**: Studies that describe the subsystem in charge of Artificial Intelligence processing. This may include scripting, strategic/behavioural (i.e., how entities decide what actions to execute) and tactical (i.e., how entities decide how to execute actions) logic, some of which may be title-specific;
- **control**: Studies that describe the subsystem in charge of orchestrating the execution of the other subsystems. It includes the game loop and other title-specific logic; and
- **hardware**: Studies that describe an abstraction layer between the other subsystems and hardware.

...
Category 2: Software architecture topics: This category shows which topics of the software architecture domain have been more applied in the development of video games. Keywords in this category are:

- **product line**: Studies that present Software Product Lines (SPLs) for the development of video games. SPLs were born as an approach to develop a diversity of software products and software-intensive systems at lower costs, in shorter time, and with higher quality [14];
- **reference architecture**: Studies that propose reference architectures for the development of video game software. Reference architectures have emerged as an element that aggregates knowledge of a specific domain by means of activities and their relations. They can promote reuse of design expertise by achieving solid, well-recognized understanding of a specific domain[1];
- **framework**: Studies that describe software frameworks, which are, in short, abstracted application layers containing common code that provide generic functionality for a domain [6];
- **design pattern**: Studies that describe the use of Software Design Patterns in the development of video games. Design patterns represent reusable solutions to recurring problems in software design [7];
- **aspect**: Studies that describe the use of Aspect-Oriented Programming (AOP) for the development of video game software. AOP has arisen as a new technology to support a better SoC (Separation of Concerns) and more adequately reflects the way developers think about the system [9]. Essentially, AOP introduces a unit of modular implementation — the aspect — which has been typically used to encapsulate crosscutting concerns in software systems (i.e., concerns that are spread across or tangled with other concerns). Modularity, maintainability, and ease to write software can be achieved with AOP [11];
- **COTS**: Studies that describe the use of COTS components that can be used as libraries, tools or building blocks for the development of video games; and
- **quality**: Studies that explore the use of software architectures aiming at improving the quality of video game software.

Category 3: Research environment: This category shows whether the studies come from the industry or the academia. Keywords in this category are:

- **academia**: Studies that were conducted and published by the academia; and
- **industry**: Studies that were conducted and published by the game industry.

Category 4: Evaluation and use level: This category shows how mature the presented software architecture is and how, if at all, it was used.

- **in use**: Studies presenting the use of software architectures that were or are in use in a video game product.
- **prototype**: Studies presenting the use of software architectures that were validated through the implementation of a prototype.

Following the creation of those four categories, the primary studies were classified according to topics within them. It is important to notice that a study may fit into more than one topic per category, or not be classifiable based on the reading of its abstract, title and study keywords. Additional categories for target platform and game genre were considered but ultimately omitted for brevity and because only a few studies could be classified by those categories.

After the conclusion of these groupings, several maps were built, partitioning the relevant primary studies by categories and by year, and the most relevant of those maps are presented and discussed.

Fig. 3 shows the result for the category “Video game subsystems”, organized by year of publication. The map shows an increasing interest in researching software architectures for video games, with the substantial rise in 2009 hinting at a larger number of future studies. A constant amount of AI research over time can also be identified, as well as an increasing interest in the control, graphics, input and network subsystems. The augmented interest in the control subsystem can be linked to the newer multi-processed target systems. The appearance of the input and network subsystems can also be explained by the emergence of new technologies, such as non-conventional input systems (e.g., motion and biometric sensors) for the first, and ubiquitous networks for the latter. It is observable, therefore, that there is a tendency towards the use of software architectures to improve each of the subsystems that make up a video game. Considering how software architectures can affect the quality of software products, however, more research work can greatly contribute towards architecturally organizing the video game subsystems and improving the maturity of future video game software.
architecture topic: Frameworks, which are straightforward enough to define and develop, and help streamline the development of video games by keeping all the common problems between different titles treated and solved in a single software package. All topics do, however, remain fairly unexplored in the context of video game development, especially aspect-oriented architectures, COTS usage and reference architectures. The study of validated architectural solutions to problems specific to a game platform or genre, for example, could give birth to reference architectures for that particular platform or genre, which could drastically reduce development time and improve product quality. The study of aspect-oriented architectures for video game development, on the other hand, could help developers model cross-cutting concerns. There is a need for discussions evaluating the use of aspects in game development, be it positively, e.g., identifying important cross-cutting concerns, or negatively, e.g., in respect to program performance or the lack of justifiable reason for the use of aspects.

Fig. 5 condenses two maps. The first crosses two categories, “Research environment” and “Video game subsystems”, and shows that the great majority of studies comes from the academia. These academic studies focus mainly on the AI, control and graphics subsystems. We have found the lack of publications originated in the industry disturbing, since game developers and software engineers alike could greatly benefit from the knowledge of how real-world problems are solved in production environments. A deeper, more detailed look into the knowledge created by the industry could greatly improve our perception of how video games evolve and how they influence computing and engineering in general, as well as contribute new ideas on how to solve complex architectural problems in software.

The second map in Fig. 5 crosses the categories “Evaluation and use level” and “Video game subsystems”, showing that most investigations involve the development of prototypes. The studies resulting in “in-use” products are about the AI, audio and graphics subsystems, and studies that result neither in prototypes nor in products are related to the AI and graphics subsystems. Although the evaluation by prototype is better than no evaluation at all, the small number of “in-use” studies can be a symptom that either the solutions proposed are not practical, or that the studies are not being taken seriously enough.

In regards to RQ2, which asks who are the authors researching the use of software architectures in video game development, no author appeared in more than one of the relevant primary studies. Therefore, it is observed that there is a lack of research groups investigating and publishing work about software architectures in the context of video game development.

IV. DISCUSSION

The use of systematic mapping to elicit previous and current research on the use of software architectures for video game development has uncovered evidence of the growing interest in this subject. This section includes the discussion of a few issues, before a presentation of limitations and lessons learned.

During Section III, all research questions established for our systematic mapping were successfully answered. This suggests that the general knowledge of software architectures in video game development has been mapped. It is also believed that
the results presented in this work are representative of the whole video game development domain, since the systematic mapping technique provides mechanisms and results that allow us to make such statement.

Considering the information gathered throughout this work, it is possible to identify interesting and important research lines that can be investigated in future work.

Current software engineering trends point towards the use of design patterns, COTS components, product line engineering, reference and aspect-oriented architectures to further reduce production costs, project risks and help build products with better quality. These topics do, however, deserve more attention from the game development community if a consensus on their value for developing video games is to be reached.

From the video game perspective, there is room for research in software architectures for all subsystems, in special the control, graphics, input and network subsystems, which have been increasing in popularity.

Also, the game development community can strongly benefit from a detailed description of industry practices that are tested and already known to be successful in real-world applications.

Regarding the limitations of this work, other categories could be established and, consequently, different related maps could be drawn. Also, new research questions could be asked, targeting specific research topics.

V. Conclusions

This work’s main contribution is the presentation of an encompassing view of the application of software architectures in the development of video games. To establish this view, a set of steps provided by the Systematic Mapping technique was systematically applied. It is believed that this overview can contribute to the video game development area, shedding light on the existing domain and presenting new opportunities for investigation.

It is widely accepted that software architectures play a major role in determining system quality. More work bridging software architecture and video game development can reduce the gap between these two areas and both improve the quality of video games and bring new architectural styles and solutions to the software architecture domain.

Motivated by the promising results, we intend — as future work — to conduct Systematic Mapping studies involving more specific topics. Thus, our research group strives towards the achievement of an even better understanding of the intersection between software architectures and video game development.

Acknowledgements

Thanks to Alexandre R. Inforzato for linguistics support. Also to Daniel Feitosa and Lucas Bueno Ruas de Oliveira for guidance with the Systematic Mapping technique.

References


