INFORMATION SYSTEMS DEVELOPMENT PROJECTS
AND THE SIMPT 2.0 CASE: HOW TO TURN
AN ANTICIPATED FAILURE INTO A SUCCESS

Tommaso Federici¹, Umberto Crisalli², Danilo Scerbo³ and Alessio Maria Bracchini¹
¹ Department of Economics and Business (DEIM), University of Tuscia, Viterbo, Italy
² Department of Enterprise Engineering, “Tor Vergata” University, Rome, Italy
³ Directorate-General for Development of territory, programming and international projects, Ministry of Infrastructures
  and Transports, Rome, Italy

ABSTRACT
This paper discusses the experience of the project for upgrading and extending a DSS of huge extent and complexity,
devoted to transport management, which is still a neglected domain for IS research. This DSS belongs to an Italian
Ministry and the project had the form of a public contract. For many reasons, this project was long-lasting and much
troubled, and when approaching the testing phase it could have been considered failed. A case investigation through an
action research reveals that drastic changes in project organization (actors composition and interaction), and in
methodology (from waterfall to agile-like methods), adopted even in this late phase of the development process,
contributed to turn an anticipated project failure into a success.

KEYWORDS
Project failures, Technical experts, Final users, Agile development, Transport IS, DSS development

1. INTRODUCTION

According to the estimates of the Project Management Institute, one fifth of the world gross domestic product
in 2009 was realized within the context of activities organized in projects. A project is a temporary
organizational effort put in place to achieve a specified goal within the boundaries of a given amount of time
and financial resources (cfr. ISO/IEC 12207, 2008 p. 5). Given the relevance and the importance of projects,
specific guidelines and collection of best practices were developed over time to provide guidance to
organizations managing projects (Wilkin and Chenhall, 2010). Notwithstanding significant efforts in the area
of project management, projects failures still occur. A project is usually considered failed if it fails to: satisfy
the requirement (quality failure), respect time limit (schedule failure), or respect the financial resources limit
(financial failure) (Anda, Sjøberg, & Mockus, 2009; Kappelman, McKeeman, & Zhang, 2006).

Estimating project schedules and costs in advance is anyhow recognized to be always difficult, and
overpassing time and costs limits is frequent in projects. With the previous criteria almost all projects can be
considered failed. More broadly then, a project is considered failed if it does not generate customer
satisfaction, or value for its customer (Savolainen, Ahonen, & Richardson, 2012). Information systems
development projects are not exempt to the occurrence of failures, and project in information systems have
indeed a long history of failing (De Bakker, Boonstra, & Wortmann, 2010).

In the problem domain of Information Systems (IS) projects successes and failures, this paper discusses
the case of the development project of a Decision Support System (DSS) for multimodal transport analysis
and simulation that was originally promoted by the Italian Ministry of Transport. The paper describes the set
of events occurred in the development project of such IS, and discusses the actions undertaken to turn an
anticipated failure into a success. The presented case shows multiple sources of interest. At first it involves
the domain of IS applied to transport management, which is still neglected in IS research. As a matter of
example, a query in the AIS Electronic Library with the keywords “transport management” in titles and
abstracts produces only 8 results that only marginally discuss the topic. The same query run for the MIS
Quarterly, Information Systems Research, and the European Journal of Information Systems produces no useful results. Other elements of interest of the case are: the huge dimension of the software to be updated, and the peculiarity of the project development that, for many causes, was reaching a failure but eventually, during the testing phase, was re-started with a different structure and then turned into a success.

The reminder of the paper is structured as follows. In section 2 the theoretical framework of the paper is described. Section 3 details the methodology followed in the paper and the unit of analysis. Section 4 reports on the case experience, on the events that occurred, and on all the actions that were taken by subjects involved in the project to turn the anticipated project failure into a success. The results of the project are discussed in section 5. Some final remarks eventually conclude the paper in section 6.

2. RESEARCH METHODOLOGY

Under a methodological perspective, this paper follows an action research framework. According to Jönsson (1991) and to Baskerville and Myers (2004), in action research projects researchers cooperate with domain experts to identify solutions to practical problems, extending at the same time their scientific knowledge. Action research is a practical oriented research methodology that perfectly suits information systems studies in real world contexts (Avison, Baskerville & Myers, 2001).

For its characteristics action research is a good methodological framework for research projects in which: (a) researchers are actively involved in solving practical projects activities, (b) the benefits from such actions are expected both for the researchers and for the company or the organization running the project or the business, (c) the knowledge obtained during the research activities can immediately be applied in practice, and (d) the research process bounds together theory and practice (Baskerville & Wood-Harper, 1996; Baskerville, 1999; Rapoport, 1970). Action research is best used as a methodological framework for those research projects in which researchers actively tackle real life problems with problem solving activities. This is exactly the case of the project investigated and discussed in this research paper where three out of the four authors of the paper were involved in the problem solving activities in the project activities.

To report the results of the action research project in the following section, we used the groups of actors in the development projects described by Boudreau and Robey (2005). Adapting their groups, in the following sections actors will be therefore grouped in the following three categories: project leaders, technology experts, and final users.

The unit of analysis of this action research study is the project for upgrading the SIMPT – Sistema Informativo per il Monitoraggio e la Pianificazione dei Trasporti (IS for Transportation Monitoring and Planning), in order to realize a new SIMPT 2.0. Such project was run under the form of a public contract and lasted eight years and nine months. Because of the changes that took place during this long period (July 2003 - March 2012), mainly in the Ministry structure, and in the external context, the project involved several people in each different role: at least 25 persons took part in a front-row activity for either the entire project duration or a shorter period.

3. THEORETICAL FRAMEWORK

As already mentioned in the introduction, failures in project activities are not exceptional events, and especially projects that develop IT artefacts (i.e. pieces of software or information systems) have a long history of failures. In IT development projects specific methodologies are commonly employed to coordinate the action of the people involved in the project, in order to ensure certainty of the outcome. These methodologies help in controlling the evolution of software development processes, allowing project leaders to plan the execution of the activities and to control their evolution. Such methodologies shall ensure that project activities achieve the specified goal, in terms of requirement that have to be implemented in the final product, in terms of costs for the development effort, and in terms of time necessary for the final product to be delivered.

Software development methodologies

Over time different software development methodologies, with their pros and cons, were proposed and applied in practice. The traditional organization of software development projects follows the so-called
waterfall method, where seven phases (requirements specification, design, implementation, integration, testing and debugging, installation, and maintenance) guide the project team from the identification of users needs to the installation of the final product in end users’ environment (Royce, 1987). This methodology received many critiques since it is highly unlikely that in practice each phase could actually be over before the next one starts. This methodology is anyhow still used in practice for its simplicity, especially in those cases where, for specific project needs like in the case of public financed projects, there is the necessity of relying on an established date for the project conclusion and certainty of project costs.

An evolution of the waterfall model is the incremental model. In the incremental model the seven phases of the waterfall model are now applied incrementally, and are iterated more times. Each time the phases are iterated, a small increment in the functionalities and in the features of the software to be developed is released (Pressman, 2010). This model tries to reduce the risk of creating software that does not suit customers’ needs due to poor interaction with him, breaking down the development process in several smaller development efforts, each one aiming at producing a small portion of the final software functionalities.

Somehow similarly to the incremental model, the prototypal model makes use of a software prototype that evolves in the development process following the requirements of the customer. In the prototypal model the phases of requirements identification, development, review, and prototype revise and enhance are iterated. At the end of each iteration the prototype is proved and discussed with the customers and further requirements are identified. With several repetitions of these process, a set of subsequent prototypes guide the project team in the interaction with the customer and in the development of the final product (Smith, 1991).

A combination of iterative design and incremental build model is used then in the iterative and incremental development model (Larman, 2003). This model is structured as a plan-do-check-act cycle since it involves iterations in the design where requirements are gathered, analysed, implemented, and eventually evaluated. Each iteration of these phases is based on the outcomes of the previous iteration. For each iteration a small increment of the software is realized through specific development activities. In this way this model mixes both incremental design and incremental development in an iterative process.

Iterative and incremental development is at the basis of the agile software development methodologies. According to the principles described in the Agile Manifesto (Beck et al., 2001), in such process both the requirements and the software evolve through the time, thanks to the results of the collaboration between small cross-functional teams. In such processes also the customer collaborates in the development activities, and the focus is on the capability to respond to changes rather than to execute a plan.

**Transport DSS**

Transportation systems are intrinsically complex as they are made up of many elements non-linearly influencing each other, and with many feedback cycles. Only some elements in the system are “technical” (vehicles, infrastructures, etc.) and governed by the laws of physics. Transportation systems functionality and their performances are usually related to transportation demand and users’ behaviour. This implies that the consequences of transportation projects cannot be predicted on the basis of pure experience and intuition, but the large number of design variables and the complexity of their interactions often require models and algorithms capable of simulating the effects of several combinations of such variables to help the designer to find satisfactory combinations (Cascetta, 2009).

A decision support system (DSS) is a computer-based information system that supports decision-making. DSSs serve the management, operations, and planning levels of an organization and help to make decisions, which may be rapidly changing and not easily specified in advance. Decision support systems can be either fully computerized, pure human based, or a combination of both (Laudon and Laudon, 2002).

The theoretical foundation of a DSS for transportation consists of a set of assumptions and a limited number of functional relationships. This paradigm represents in an abstract way transportation services and their performances (supply model), travel demand and behaviour of system users (demand model) as well as their interactions (demand/supply interaction model). Apart from the internal complexity, transportation systems are closely interrelated with other systems, which represent the “rest of the world”. Transport projects may have implications for the economy, the location and intensity of the activities in a given area, the environment, the quality of life and social cohesion. For this reason, the estimation of effects and impacts is becoming increasingly important in the evaluation of transport interventions.

Moreover, the expansion of national transport policies, ranging from strategic planning of infrastructure investments to their management, with the focus primarily on efficiency, environment, security and regional equity, requires the adoption of advanced analytical tools and more sensitive to the national policy choices.

The need to make use of DSS for transport planning at the national level is common in many European
countries and it has been receiving increasing attention during recent decades (Button, 2000). For a concise state-of-the art on DSS to support national transport policies, the reader can start from Lundqvist and Mattsson (2001) which reports the national DSS developed, under-development and/or being updated for the following countries: Belgium, Denmark, Germany, Hungary, Italy, Netherlands, Norway, Sweden and the United Kingdom, among which The Netherlands and the United Kingdom have been the pioneers.

4. CASE DESCRIPTION

Since 1993 the Italian government started the development of the Italian DSS to support transportation policies called SIMPT. In the sphere of the theoretical framework of transportation DSS pictured in section 3, it is clear the importance for a country like Italy to take advantage of a tool that supports quantitatively the complex process of policy-making. The SIMPT should represent in Italy the tool to carry out the analysis of mobility at the governmental level. The evaluation process must support all the stages of the transport plan development: the preliminary analysis, the consistency checks, and the analyses of the effects of actions in relation to the objectives of efficiency, safety and sustainability. This process should be supported through the use of a diverse and scientifically based analysis tools (including quantitative methods, models and algorithms), which will ensure transparency and correctness in the development of the whole decision process. The reader should note that DSS for national transport planning is not only a government interest, but it could also be used as a specific in-depth marketing tool by national transportation companies both private and public. For example, the Italian case reports at least two transportation companies that own their specific DSS for transport planning: the DSS developed by the Italian National Roads Corporation (ANAS), and the one developed by the Italian National Railways (Crisalli, 1999). The functional architecture of a transportation DSS like SIMPT is based on:

A database, which stores the input data (i.e. the characteristics of supply and demand) and output (as the results of the assignment and the performance indicators) relating to the transport system;

A system of models, which represents the core component of the DSS: there are models of supply and demand that interact among each other through the simulation tool that provides functional variables which define the transportation scenario, such as, for example, the vehicle flows, the level of service attributes, the modal share, and so on.

The user-system interaction functions (namely the graphic user interface, GUI): it is especially important since it allows the use of a complex system such as the DSS also to non-expert users. In particular it allows creating a scenario of the project, the definition and specification of transport components such as the transport demand and the service timetables, as well as the analysis of simulation results.

In the way it was designed, SIMPT allows the user to formulate intervention hypotheses on the supply configuration of the transport system and select socioeconomic and demographic scenarios in an intuitive way, freeing the user from the need to manage specific information. It is in fact the system that governs the coherence and consistency of the data, possibly making further inquiries to the user. The simulation process is mainly divided into two distinct phases: the construction of the simulation scenarios, and the activation of calculation functions. For what concern the SIMPT modelling system, there are:

Demand models, for both passengers and freight, which allow the simulation of the main characteristics of the transportation demand;

Supply models, for both passengers and freight, which support the representation of the main features of the infrastructure and services of the (multimodal) transport system;

Demand-supply interaction models (also known as assignment models), that allow to estimate the use of infrastructure and services for both passenger and freight;

Models for the analysis of impacts and performances;

Models for the analysis of costs and revenues.

The models in SIMPT differs from other national models that can be found in the literature (Button, 2000; Lundqvist & Mattsson, 2001) for some features: the special attention on freight modelling and their integration with passenger models into a single modelling framework, the differentiation of models depending on the season (summer models and winter models), and by day of the week (models for weekdays and models for holidays), the distinction between the travel patterns of residents (nationals) with models of foreign exchange, particularly relevant given the rich tourism industry in Italy, and finally a sophisticated
treatment of the path choice for road transport, by considering separately the interregional routes from the long distance ones. By contrast, the Italian modal choice models are, however, less detailed than other national models. This aspect has been improved in the latest updates of the system.

After ten years since its first development, in 2002 the Directorate-General for the Programming (DGP) of the Italian Ministry of Infrastructure and Transports in charge of the SIMPT decided to promote a project for upgrading and extending the original system (conventionally named since then SIMPT 0). Many reasons of different nature were behind this choice: the strict exigency of updating the operating environment, particularly the hardware; the request for some new, user-friendly, functions; the opportunity of evolving some of the embedded transport models. About the first issue, the huge machine resources required for the simulation computation had oriented the first developers in 1994 to adopt a customized server, which later became really difficult and expensive to be maintained. This also impeded any upgrading of operating systems, DBMS, and application software, because the more recent versions did not work on that machine.

The Ministry invited tenders with a public call which involved thirteen activities addressed to system upgrading, and other thirteen activities, mainly devoted to the collection of data (through surveys, interviews, and databases acquisition), essential to ground simulations on updated data. The contract was assigned to a temporary association of companies, with qualified experiences in transportation IS. At that time, the dimension of the SIMPT as a software was very great, both in terms of lines of code (more than 250,000), and of function points (about 6,300). A very rich library of documents (more than 1,200 pages) complemented the system, describing in detail all the transportation models. Technical maintenance documents (excluding the database) were missing, like for other software developed in the very years.

Because of the many organizational changes intervened in the Ministry and in the DGP, the project started one year later than expected, under a new Director and with, as final users, persons who had never used the SIMPT 0 before. The Director chose then to create a panel of scientists of transport engineers and in IS, who already knew the SIMPT and DSS in general. Their tasks were to support his decision making, but also to guide the developers by providing them technical needs and explanations. Considering these difficulties and the complexity of project activities it was firstly decided to concentrate the project on data collection. During this first phase (July 2003 - June 2006) the activities concerning the upgrading and extending of SIMPT 0, planned following the traditional waterfall method, were considered as marginal. Developers concentrated on requirements analysis, particularly on the definition of the transportation models to create or adapt.

Later on, when data collection activities were almost over, but development was still at the beginning, the Ministry was split up in two different Ministries (respectively of Infrastructure and of Transport). For about one year the project did not realize substantial progresses, while expecting new responsibilities assignments. Meanwhile the panel of scientist was not renewed and the developers remained without expert interlocutors for their work. The head of the Directorate-General in charge of SIMPT changed more times in the subsequent period, with a consequent stop and go effect on project activities. To conclude the project, developers proceeded however with the development activities, without neither formalized requirements nor a guidance as regards technical aspects, making use of the little information already gathered, and of the knowledge gained in using the SIMPT 0. The presumed final version of the system, still to be tested, was eventually released to the customer Ministry (that in the meantime was newly re-joined) at the end of 2009.

The Ministry, following the law prescription, established a Test Committee (TC), formed by an internal manager as committee chief, and two external experts (one in transportation, and one in IS), who already were member of the former panel of scientist and soundly knew the SIMPT. The TC task was that of providing, after an adequate tests campaign, a judgement of acceptance, either positive or negative, of the system released. Just at the very initial checks the software, even though apparently complete in terms of expected functionalities, presented severe issues, both in terms of system requirements (i.e. installation did not end successfully, functions were not correctly assigned to client and server), than in functions’ use (many of them returned errors or did not accomplish their aim). Because of these problems, it was not possible to proceed to more detailed tests neither to assess the quality of data produced with simulations, which in the end is the main purpose of a DSS. The problems appeared too big to be solved performing a traditional check and bug-fix phase, and the project had to be considered failed, at least as regards development activities.

When such considerations were reported to the DGP, it was decided not to close the project as failed. An exploration of any possible solution to obtain software that could correspond to the originally aims was started. This decision was made for three reasons: the long-lasting and costly litigation between the Ministry and the developers which probably would have started, the consciousness of the Ministry of the problems created to the developers, and overall the real need of the Ministry of a well performing DSS for transport
planning, in order to accomplish its institutional aims. Then, once verified the willing of the associated companies to rework this part of the project, the test phase was redrawn. Being it impossible to restart with the former waterfall method, a new method was proposed to the developers. The experts of the TC would have played the role of final users, with a specific commitment by the DGP to act on its behalf towards the developers. A closer collaboration was established by the two sides, based on periodical meeting in front of a working software release, and on a continuous bi-directional exchange of information. Even more important, it was decided to proceed by releasing subsequent small software upgrades, each one solving a coherent subset of issues. To support this plan, the test cases initially designed by the experts of the TC were detailed and adapted in order to represent what final users would have expected by the different functions and/or in diverse situations by the SIMPT 2.0.

An iterative cycle then started with subsequent steps designed as follows: the experts of TC performed a subset of tests on the current software release starting from the simpler functions; the revealed bugs and not compliances were reported in an interactive session to the developers; a new release which possibly solved the issues found was delivered by the developers; then, once verified the solution to what reported in previous tests, new tests were performed. In the first version the test cases were twenty, mainly regarding basic functions like install, log in, data access, and data change. These test cases evolved with time and grew up to fifty, later including twenty-two simulations of different transport scenarios that asked for specific testing procedures.

With regard to the contents and reliability of the SIMPT, test cases aimed at validating the correct operation of the GUI, the procedures for database management and the transport modelling system. At the first execution, and sometimes even in the following ones, some test cases succeeded, some returned a bug, others showed an interface or process designed more to a high level expert than to a Ministry's user. As regards simulations, given the complexity of the transport system described above, it is not possible to verify the output of the system in terms of analytical comparison of simulated values with target ones (as usually done for other engineering problems). Instead, the response of the modelling system has to be evaluated through the variations of the transportation variables (e.g., some characteristics of supply or demand) between the simulated and the actual scenarios. For this reason, the DSS was tested on the basis of an extensive series of tests, which have affected the response of the DSS to changes in freight transport (e.g., increasing the cost of road transport), in passenger transport (e.g., introduction of new rail high speed), as well as through the assessment of network flows carried out by the assignment of the transport demand to the national network for different simulation scenarios, including the simulation of traffic congestion.

Also considering the extent of the SIMPT and the complexity of most of its functions, to achieve bug-free software, compliant with the contract statements and, overall, with the Ministry's needs, twenty-one iterations were eventually performed. As last step, all the designed test cases were newly performed just with the intent of a test session. Finally, it is also to remark that, because the long-lasting period passed since the call for tender, no more being possible to operate on the very old versions of the operating environment there prescribed, this last phase provided as an externality the update of SIMPT 2.0 to the state-of-art of server and client operating environments.

5. DISCUSSION

The project for the upgrading and the extending of SIMPT shows a really troubled history, along which development methodology, people, responsibilities and even the same customer structure changed. To better interpret the project history, we have adapted the groups of actors in the development projects described by Boudreau and Robey (2005), because they can help to make order in the project scene, also with the aim to draw some hint out of the case. Following these authors, we have categorized the people intervening in the project into three groups: project leaders, technology experts, and final users. Members of the first group are persons that lead the project activities, usually with a strong commitment over it. Members of the second group have sound competences on technology, usually the internal or external developers' team, sometime sided by other experts. The last group gathers persons who have competences on the specific domain where the IS has to operate, and who will actually use it.

To fully accomplish its aims, a project should not only see the three groups adequately play their role, but it should also encourage a continuous and positive interaction between technological experts and final users.
The information exchange put in place between these two groups of actors generates new, essential, knowledge for both of them. Technological experts acquire, through the interaction with final users, specific knowledge on the domain, and also on users' need, way of interact with systems, expectations. On the other hand, final users become more informed on how technology works and on what it may really offer, coming also to improve and better specify their requests.

Table 1. Actors operating during the project phases.

<table>
<thead>
<tr>
<th>Project phase</th>
<th>Project Leaders</th>
<th>Technical experts</th>
<th>Final users</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Requirements specification</td>
<td>Director of DGP</td>
<td>Developers team</td>
<td>Panel of scientists (only partially)</td>
</tr>
<tr>
<td>2. Design</td>
<td>Diverse managers</td>
<td>Developers team</td>
<td></td>
</tr>
<tr>
<td>3. Development</td>
<td>Diverse managers</td>
<td>Developers team</td>
<td></td>
</tr>
<tr>
<td>4. Testing (beginning)</td>
<td>TC chief</td>
<td>Developers team</td>
<td></td>
</tr>
<tr>
<td>5. Testing (iterative cycle)</td>
<td>TC chief</td>
<td>Developers team</td>
<td>TC experts</td>
</tr>
</tbody>
</table>

Table 1 shows the actors that played the relevant role along the project phases. As it can be noticed, a real, collaborative, fully informational exchange between the technological experts (in this case the members of the developers team provided by the associated companies) and the final users started only in the last phase of the project, when it was decided to redesign the project organization. Actually, in the previous phases the development project missed final users' component, because in the DGP there was no employee who already knew the SIMPT, while the scientists of the panel were not designed to play such role, and had however few occasions to do it. Despite the high level of transport IS competences of the developers, their lack of knowledge on the SIMPT, mixed with the absence of interaction with informed users, drove to a system that resulted both unusable and different from consumer's expectations. The situation dramatically changed when it was decided that the TC experts, who already deeply knew the SIMPT, would have had to play the final users' role, by transferring information to the developers and guiding them ("this interface is designed to be used by a research centre engineer, not by a Ministry employee").

Another important element to be highlighted, in order to identify what helped to turn the project destiny, is the decided change of the development methodology. The waterfall method, initially adopted like in other public projects, could not be replicated after six years. Moreover, two other conditions supported the choice to adopt a different method: the availability of a certain final date, necessary when signing a public contract, was any longer as much important, and more significant, even though not compliant and with many bugs, the released system could be used as a prototype to be assessed and improved.

Although never named in this way by the project members, the methodology adopted in the final phase presents several elements that recall the agile methodology (Pikkarainen et al., 2005). First of all, the already mentioned start of a continuous, open collaboration between customers (in this case, the two TC experts on behalf of DGP) and developers. Second, the use of the released system as software to work on to provide requests of changes and improvements. Third, the adoption of iterations, each one involving a small subset of functions to be changed, fixed or realized, and then a small planning for each iteration. Fourth, the recourse to face-to-face sessions with TC experts and developers in front of a same computer running the new SIMPT, in order to assess the new release, and in the last period also to activate the pair programming technique. Fifth, because of the circumstances, the forcibly adoption of test-driven development, which more focuses the development on what the user really wants. All this points seem to be as many drivers for the project recovery. They also seem to testify the successful use of the agile methodology even in a complex project, which laid in a problematic situation. As regards the high number of performed iterations, it does not have to astonish, when considering the extent and the complexity of the system in question.

6. CONCLUSION

This paper investigates, through an action research, the project for upgrading and extending the SIMPT, a large and complex DSS devoted to transport monitoring and simulations. As the system belongs to an Italian Ministry, the project had the form of a public contract. Even if this case is peculiar for its characteristics, it
can provide very useful generalizable hints. For several reasons this project has been long lasting and much troubled, and when approaching the testing phase it could have been considered as failed. Some changes in the project organization and in the methodology put in place even in this late phase of the development process contributed to turn a nearly failure into a success. From the discussion of the case, reinforcing final users’ roles, and adopting methods similar to those of agile development, appear to have been the most effective measures to turn the failure into a success.

REFERENCES


