Euro Truck Simulator 2 Reverse Engineered Requirements Document

Omar Asadi

Aftab Hussain

University of California, Irvine

TABLE OF CONTENTS

1. Overview

In this work we reverse-engineer the elaborate requirements of the game, Euro Truck Simulator 2 (ETS2), available at [1]. ETS2 was released by SCS Software, on October 2012. The fundamental gameplay constitutes of driving trucks around depicted European cities for delivering cargo at various locations. It keeps a track record of the player's completed tasks and offers career progression for the player in terms of income. The game has been widely popular[1], with high rates of downloads and is available for purchase at the multi-player online gaming platform, Steam[2].

Reverse engineering the requirements document of a given system is a challenging task particularly for large legacy systems which are deployed in a distributed manner. The main difficulty this task poses is in capturing the complexities of all components of the system in a coherent and non-conflicting fashion. A way to address this difficulty is to build the requirements models of the system in an incremental way [3], an approach we tried to follow while doing the same for ETS2.

We now present the organization of our report, which also reflects the work flow of this reverse engineering project: In Section 2, we present a description of all the relevant stakeholders of the game. In Section 3, we present a case study of the game with respect to its gameplay. In Section 4, we present the models of the system. In Sections 5 and 6, we give a description of the techniques and tools used, respectively. We conclude the report in Section 7, mentioning challenges addressed during the work.

2. Stakeholders

We have segregated the stakeholders of ETS2 into two spectrums: Marketing and Investment (Section 2.1) and Game Construction (Section 2.2). While the former enlists all stakeholders that relates to fulfilling the business requirements of the game (which are relevant towards the development of any software product), the latter comprises of those that are connected to the actual development of the game. It should be mentioned that this categorization should not be considered as a strict separation of concerns as some stakeholders, e.g. executives, will have roles to play in both the construction and the marketing and investment of the game. This segregation is solely to provide a grouping of the stakeholders as per the perceived characteristic of the majority of each stakeholder's responsibilities.

2.1. Marketing and Investment Stakeholders

Executives. They are the chief-category officers of the game vendor company who are the key drivers of the project in terms of providing the vision, approving resource utilization, marketing the product, and maintaining investor and shareholder relations.

Investors. The investors allocate capital to a potentially profitable project. They provide the crucial funds that may be allocated appropriately for the development of the game.

[1] The game has sold over 500,000 units as of April 2014, http://en.wikipedia.org/wiki/Euro_Truck_Simulator_2

[2] http://store.steampowered.com/

Shareholders. The shareholders are the owners of the company and thus the profits of the vendor company has a direct impact on their interests. They may have voting rights over membership in the board of directors, and thus it is important that any new decision to go-ahead in the development of the game, and the outcome of the sales of the game plays positively to the shareholders.

PR Team. They are responsible for marketing the product, branding, media relations, creating and maintaining awareness of the Euro Truck product line, communicating with advertising agencies, and receiving customer feedback via different media for the purpose of shaping the story of the product.

2.2. Game Construction Stakeholders

Product Managers. Assign project to development teams, monitor the work flow of the project, set milestones, and meet software architects in order to get their progress feedback and address their resource concerns.

Software Architects. These are people who will generally act as team leads of different components of the project: for example, user interface component, networking functionalities, gameplay logic. Each team will generally comprise of 3-5 developers.

QA Team. The Quality Assurance Team's main focus is to verify and validate the development of the different teams of the project, iteratively. Their roles include preparation of test data for each phase of the project, testing, etc.

Developers. They are the core workers of the project, who are responsible for coding and implementation of the entire constructs of the game. They will work in different components of the project under the supervision of a software architect.

UX Designers. They provide the general layout of the game, the designs of the menu pages, etc.

CAD Engineers. ETS2 being a simulation game would require sufficient realism in the designs of the trucks and cities. CAD engineers would fill in that role by creating 3D visualizations of those real-world entities.

Legal Experts. Their job would be to ensure no copyrights/governmental laws are violated during the gathering real-life information (city plans, city landscape, truck designs, etc.) which would be necessary to develop various entities of the game, check for any preexisting patents belonging to other parties which constitute ETS2's ideas, and to mediate the exchange of rights when necessary.

3. Case Study

This interactive real-time software allows the player to simulate the experience of being a truck driver for a European trucking company. The system goes into incredible detail, including but not limited to:

1. The complexity of the driving controls

   a. The number of driving functions under the player's control
      i. Automatic transmission handles the gear shifts for the player
      ii. Manual transmission requires the player to shift gears at the appropriate time in order to optimize speed

   b. The type of player input (keyboard, mouse and keyboard, or steering wheel controller)
      i. Keyboard only is the simplest: up, down, left, right to correspond to forward, brakes/reverse, turn wheels left, and turn wheels right, respectively
      ii. Mouse and keyboard uses the keyboard for acceleration/brakes and the mouse for direction
      iii. Steering wheel offers the most authentic driving experience

2. The city in which to drive

   a. Choose from cities all over Europe
   b. Each city changes both the scenery and the layout of the roads

3. The types of trucking jobs to accept and complete

   a. Short drives
   b. Long drives
   c. Timed drives

4. The difficulty of completing the trucking jobs

   a. Combination of the length of the trip and the time until the deadline

5. Damage taken by the truck or cargo

   a. The truck must be repaired in order to function properly
   b. Damage to cargo is permanent and can reduce the reward or even cause the player to fail a job
   c. Time taken to repair the truck will delay the successful completion of the job, potentially reducing the payout

Upon completing jobs, the player can advance their career and their trucking company to the point of international supremacy.

4. Models

In this section, we present the models that were derived for ETS2. We present the goal model (4.1), the system vision (4.2), and finally the non-functional requirements hierarchy (4.3).

4.1. Goal Models

The goal-oriented models illustrate the various objectives of the game. For building the goal models we have used the *i** framework (described in Section 5.1). A goal-oriented model helps in setting the foundation for requirements elicitation, driving the design of the product, and resolving conflicts [7]. Using the *i** framework, we derived two goal models of ETS2, the Strategic Dependency Model (SD), depicted in Figure 1, and the Strategic Rationale Model (SR), depicted in Figure 2.

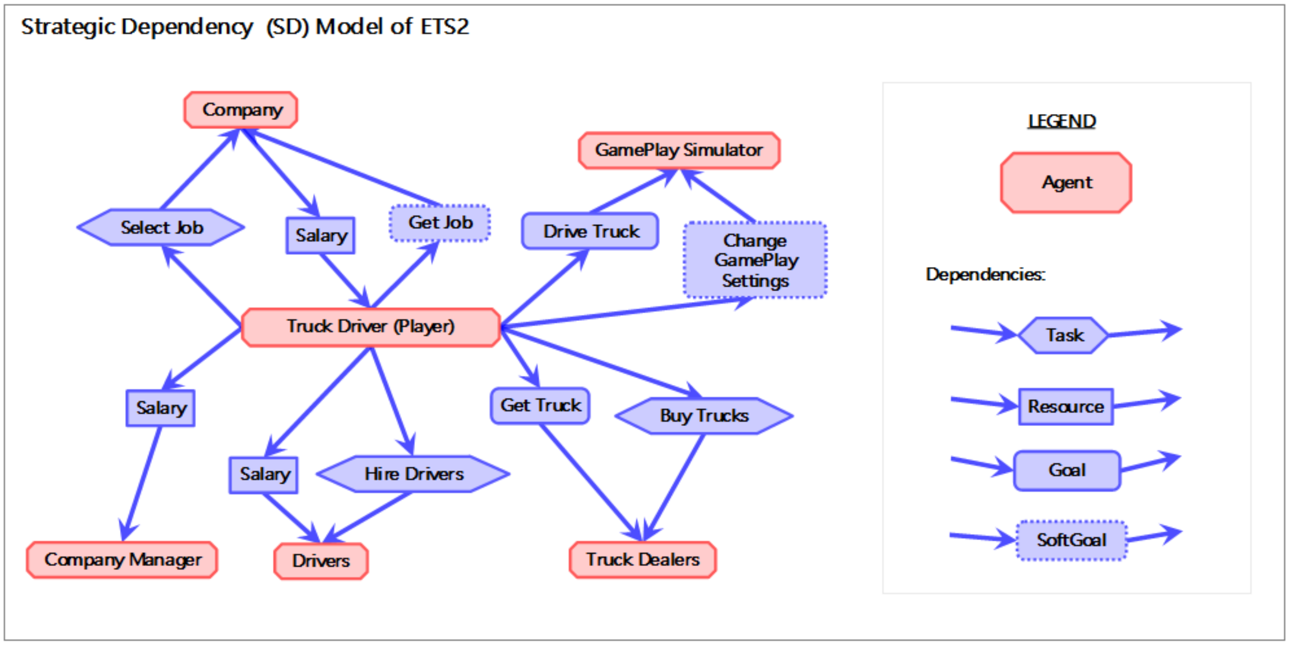

Figure 1: Strategic Dependency Model of ETS2.

The SD model portrays the interactions between the various agents of ETS2, in the context of ETS2's gameplay. The only real/tangible agents here are the truck driver (the player) and the gameplay simulator, the primary agent/actor being the player. The player has task, resource, and goal dependencies with fictitious agents such as the company, the driver's company's manager, truck dealers, and employees (other truck drivers). An example of a task dependency is that of job selection, where the driver selects a job from a list of offers from different fictitious companies. The truck driver receiving salary is an example of a resource dependency. The primary goal in this scenario is that of driving the truck, which is the main action of playing the actual simulator. A softgoal which may be associated with playing the simulator is that of changing the settings of gameplay. Since this is not an essential feature in order to play the game, we have classified it as a soft goal in the SD model. Another softgoal that was portrayed in the model is that of getting a job. This is an added feature as the game offers the player to start playing with the simulator right away, with preset default truck and location selections. From the SD model, we can thus understand the intentional relationships between the different agents.

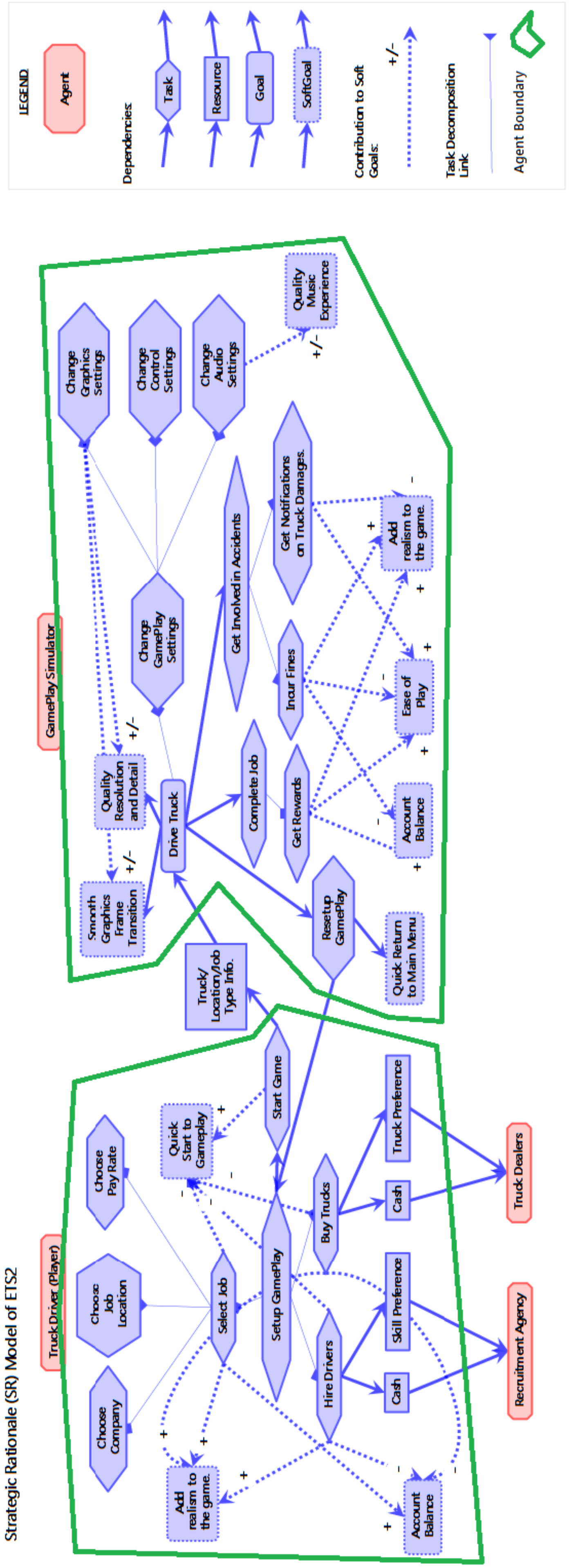

Figure 2: Strategic Rationale Model of ETS2

In the SR model we go into greater depth with regard to exploring the intentions of the agents, which is the fundamental concern of the *i** approach. As was mentioned earlier, in principle, there is only one actor in ETS2 with regard to ETS2's gameplay, which is the player. Therefore, in the SR model we have elaborated on the tasks of the player, with a focus on intention. In addition, we have done the same for the only other tangible actor within ETS2's gameplay, which is the simulator. Although, two more actors were included, the "recruitment agency" and the "truck dealers", their sole purpose was to show breadth in the capabilities of the player.

As can be seen, SR emphasizes on soft goals and how certain tasks can affect the achievement of those goals. All the soft goals have been set with respect to the expectations of the player. Consider the task of selecting a job; this task has three further sub-tasks relating to company, job location, and pay rate. This clearly contributes positively to the realism of the game, which is a desired softgoal of the game. It also adds positively to the account balance of the player within the game environment, as it allows the player to earn. However, this task negatively impacts the subgoal of "quick start to gameplay", as it could cause a delay in actually playing the game. This relation should remind developers and designers of making the job selection process as effortless and seamless as possible.

Let us consider another example of a task "start game"; this task has a direct goal dependency on the goal "drive truck" which exists within the boundary of the GamePlay Simulator. This goal has a couple of subgoals: "smooth graphics frame transition" and "quality resolution and detail". These subgoals can be affected positively/negatively by changing the graphics settings. What has not been shown in the SR model in this regard is that these subgoals have an inverse relationship. Improving one would adversely impact the other. Thus another subgoal, which could be derived from these two goals is that of finding an optimal setting of the two.

Finally let us consider the task of "get involved in accidents", which is shown to be subdivided into two more tasks of incurring fines and getting damage notifications. While incurring fines negatively affects the subgoals "account balance" and "ease of play", it contributes positively to the realism of gameplay. Getting damage notifications is shown to positively contribute to the ease of play, as it instantaneously tells the user about the current state of the truck. However, such a feature generally do not exist in real-life truck notification systems and hence it may negatively contribute to the realism of the game.

In addition to further exploring the impact on softgoals in the SR model, we have also adopted an expansive approach for analyzing soft goals in the SR model by decomposing them into smaller subtasks. For instance, "Change GamePlay Settings" is a soft goal in the SD model, but is a sub-task of the goal, "Drive Truck" in the SR model. This was done because it was necessary to explore the softgoals in the SD model in greater detail, which forced the expansion of the sub-goal into a set of tasks.

4.2. System Vision

This model presents the broad vision of the overall system that we believe were agreed upon by all stakeholders of the system based upon our investigation of the game. We used the Rich Picture Method (explained in Section 5.2) to construct the System Vision.

Figure 3, shows the system vision of the game. As can be seen, we have constructed a rich picture with the game construction perspective. In other words, we have shown all elements,

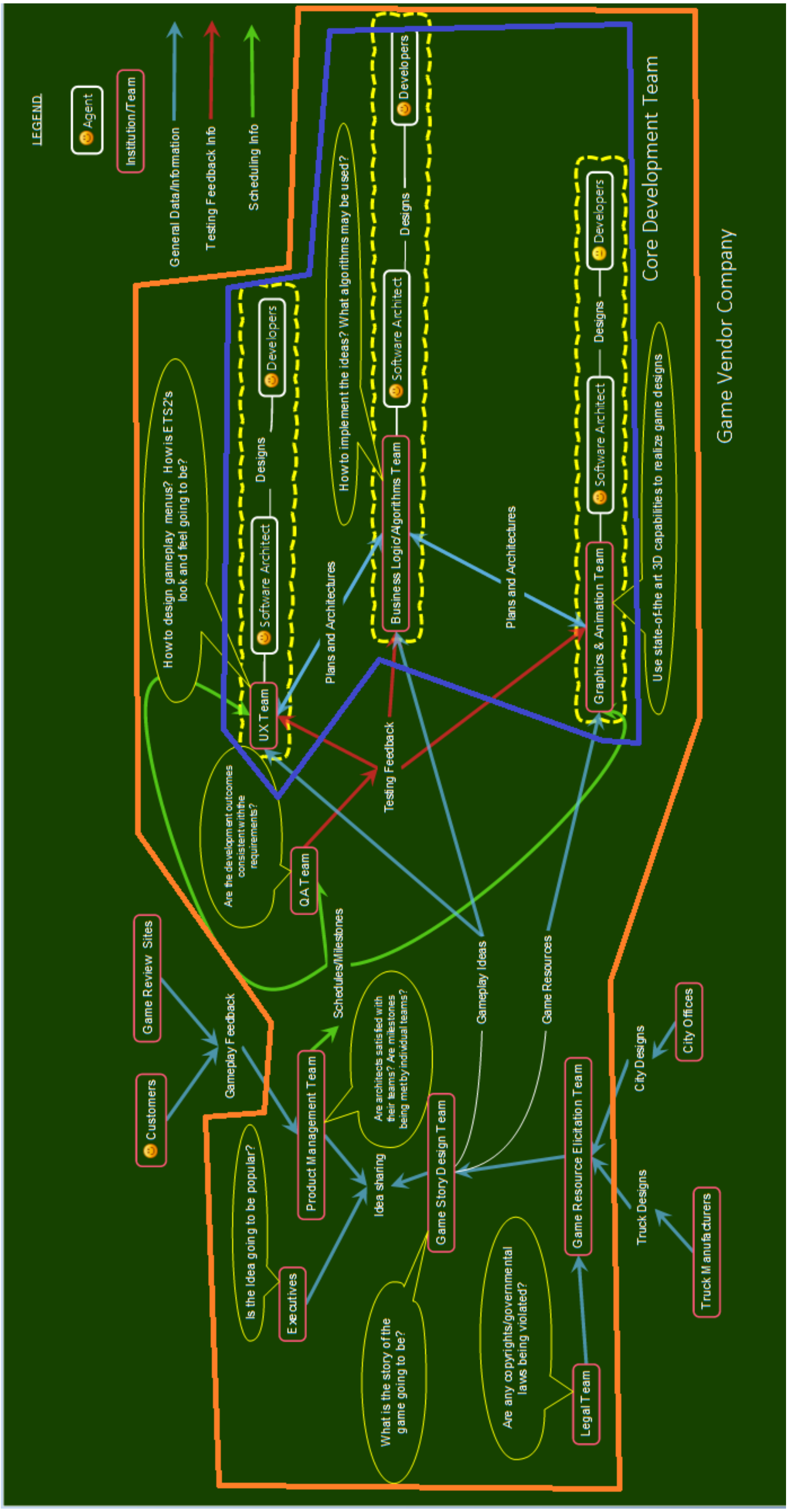
LEGEND
Agent
Institution/Team
General Data/Information
Testing Feedback Info
Scheduling Info
How to design gameplay menus? How is ETS2's look and feel going to be?
Developers
Designs
Software Architect
UX Team
Plans and Architectures
How to implement the ideas? What algorithms may be used?
Developers
Designs
Software Architect
Business Logic/Algorithms Team
Plans and Architectures
Developers
Designs
Software Architect
Graphics & Animation Team
Use state-of-the art 3D capabilities to realize game designs
Core Development Team
Game Vendor Company
Are the development outcomes consistent with the requirements?
QA Team
Testing Feedback
Schedules/Milestones
Gameplay Ideas
Game Resources
Game Review Sites
Customers
Gameplay Feedback
Product Management Team
Are architects satisfied with their teams? Are milestones being met by individual teams?
Is the Idea going to be popular?
Executives
Idea sharing
Game Story Design Team
What is the story of the game going to be?
Are any copyrights/governmental laws being violated?
Legal Team
Game Resource Elicitation Team
City Designs
City Offices
Truck Designs
Truck Manufacturers

Figure 3: System Vision of ETS2

flows, and stakeholders that are likely to be involved in the development of the game, at a high level of abstraction.

We have grouped the model's entities using 2 *structure* boundaries: the game vendor company and the sub-structure, the core development team. Further structure categorizations were not made in order to not clutter the diagram and also to avoid straying from the main theme of the model, which is game construction. Broken boundaries represent sub-teams within the core development team, and call-out symbols have been used to represent the *concerns* of individual stakeholders. The arrows represent the flow of information during the construction of the game, and thus illustrate the *process*.

The main game construction process starts with the executives, who set the goals in collaboration with the game story thinkers, while adhering as much as possible to the demands of the customers obtained from market research data provided by product managers. Based on the deliberations in this initial phase of determining of how the story is going to be, appropriate resources (possibly copyrighted) are elicited. These resources include truck designs and city designs. The process is overseen by a team of legal experts to help navigate the legal boundaries of sharing copyrighted material. The game story ideas (in the form of detailed specifications) and resources are then shared with the core development team, whose milestones are set by product managers in coordination with software architects. The outcomes of the development teams are iteratively tested by the Quality Assurance Team, who perform their spectrum of testing techniques based on test cases generated from the game specifications.

4.3. Non-Functional Requirements

The non-functional requirements of ETS2 include the ability to distribute the game via the Steam platform, the expectation that the game runs smoothly on a wide variety of different hardware platforms, and the enjoyment factor of the game by the customers (players).

The first non-functional requirement, **distribution** via Steam, presents a challenge that has almost nothing to do with the software itself. The Steam platform provides many developers and publishers the ability to reach a wide variety of gamers while also allowing gamers to consolidate and organize their purchased games in one location. This integration with the Steam platform allows the developers to almost completely offload distribution concerns to Valve Software (developers of Steam). Integrating with Steam means Valve will be the one worrying about server load and bandwidth, not SCS Software. All SCS Software needs to worry about is the interface with Steam to allow this distribution to take place.

The second non-functional requirement, **hardware-independence**, presents a unique challenge for the developers. Real-time interactive software for general-purpose PCs comes packaged as only software and required libraries and is designed to run on any machine meeting the developer-determined minimum system requirements. The alternative is software for gaming consoles like the XBox One and the PlayStation 4 where the hardware specification is known in advance and the software can be optimized for that specific combination of hardware. Gamers with general-purpose PCs typically have widely varying hardware, but they all expect the software to perform smoothly on their specific hardware setups. This requirement forces SCS Software to implement various levels of performance for their software by modifying graphical settings. The most common areas of variability are texture detail (affects graphics card RAM usage), anti-aliasing level (affects graphics card load), and shadow detail (affects graphics card load). By implementing many different levels of performance in each of these areas, the developers can ensure that any machine that meets the minimum system requirements (usually defined to passably run the software at the absolute lowest graphical settings) can smoothly run the software.

The third non-functional requirement, **enjoyability**, is potentially the most challenging requirement to meet, since it is the most subjective. Each of the previous two non-functional requirements can be accomplished by checking off items in a list. This requirement depends entirely on the customers. The developers must take care that they perform acceptance testing with a representative focus group before releasing the game to the general public. Even the best-intentioned game ideas can result in critical and commercial disasters, such as Titus Software's Superman for the Nintendo 64 console[3]. The concept seemed great at first: allow the player to fly around Metropolis as Superman, exhibiting incredible, heroic feats. The implementation, however, fell flat on its face. Many critics cited the unresponsive controls as a major downfall of the game, among other things. ETS2, on the other hand, appears to have met this requirement due to the above-average reviews and a Sim of the Year award in 2012[4].

Description of Techniques used

5.1. *i** Framework

We used this framework to construct the goal model of ETS2. The *i** framework is a modeling framework geared towards the early phase of requirements engineering [1] that adopts an agent/actor-oriented approach. In particular, it helps in mapping the dependencies among the actors in a system with respect to the organizational context and also the rationales behind those dependencies. It provides two models for this purpose, a strategic dependency (SD) model and a strategic rationale (SR) model. While the SD model mainly focusses on the dependencies, the SR model elaborates further on the concerns/interests of the stakeholders. The SR model thus reveals more about the "intentional constructs" of the SD model, i.e. the goals, tasks, and softgoals. Elaborate details of the technique could be found in [4], [5].

5.2. Rich Picture Method

The rich picture method comprises of the construction of a rich picture which captures the rationales of the work context and shows how the design can be affected [6]. The three major components of a rich picture are structure, process, and concerns. Structure refers to the more concrete and immutable aspects of a work context, for instance organizational hierarchy, which could change only gradually. Process refers to the various flows among the various entities within the system and its environment. Finally concerns, comprise of the possible individual issues of the stakeholders of the system.

[3] http://en.wikipedia.org/wiki/Superman_(1999_video_game)

[4] http://en.wikipedia.org/wiki/Euro_Truck_Simulator_2

6. Description of Tool Used

In this section, we describe MindMaple[5], the tool that was used for generating the models' illustrations presented in this report.

MindMaple is a mind-mapping software that helps in visually organizing thoughts in the form of maps. With its rich set of drawing features, like trees, relationship boundaries, etc, it is possible to intuitively convey information through the diagrams. In addition, it provides short-cut keys for drawing specific components of the model, such as call-out symbols, and relationship arrows, which helped to draw the diagrams easily and efficiently. The software is available for all major OS platforms, including Windows, Linux and Mac. Figure 4, shows a screenshot of MindMaple Lite v1.65.

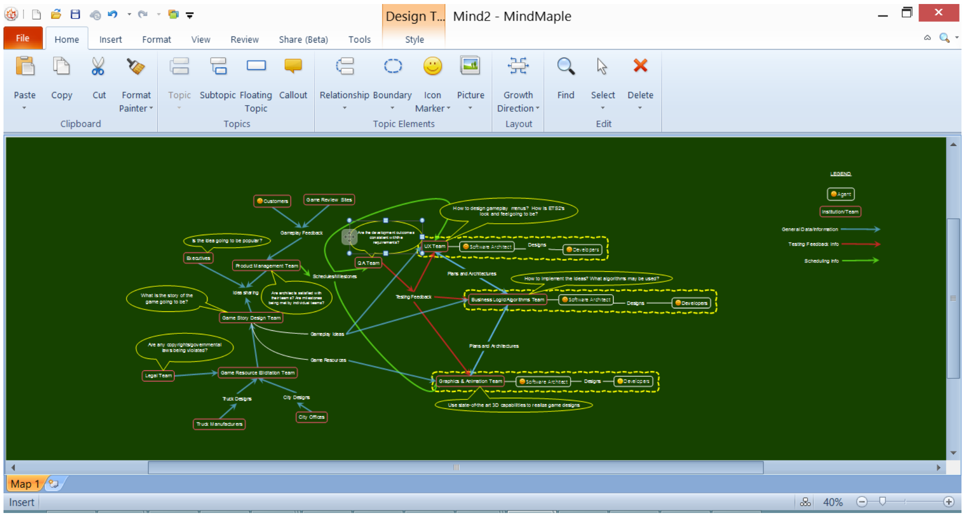

Figure 4: MindMaple

7. Conclusion

In this report, we have presented the requirements of ETS2 obtained via reverse engineering the requirements. We first described the various stakeholders of ETS2 by categorizing them into two groups based on functional perspectives. Then we derived and elaborated upon the goal, system vision, and non-functional requirements models of the game.

As was mentioned earlier, reverse engineering requirements documents of any software system is a lengthy task, which may offset the benefits it provides due to the large effort needed in building it. Among the main challenges that were faced in conducting this investigation was to ensure consistency and uniformity among the models. We tried to address this challenge by iteratively comparing the models during the construction of the models. Another challenge was in determining the scope of the project, especially when enlisting all possible stakeholders. In addressing this we followed the philosophy of focusing on interactions, tasks, and communications of the vendor side as much as possible. For instance, an investor would consult independent market research agencies for gathering information on the demand of simulator games like ETS2 before choosing to invest in the project. Since this interaction

[5] http://www.mindmaple.com/

between the investor and the market research agencies is entirely outside the purview of the vendor company, we deciding against consider market research agencies as one of the game's stakeholders.

References

[1] http://www.eurotrucksimulator2.com/

[2] Eric S. K. Yu. 1997. Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering* (RE '97). IEEE Computer Society, Washington, DC, USA, 226-.

[3] Reverse engineering requirements, http://www.blueprintsys.com/back_to_the_future_reverse_engineering_requirements/

[4] Eric Siu-Kwong Yu. 1996. *Modelling Strategic Relationships for Process Reengineering*. Ph.D. Dissertation. University of Toronto, Toronto, Ont., Canada, Canada. UMI Order No. GAXNN-02887 (Canadian dissertation).

[5] Eric S. Yu. 2009. Social Modeling and i*. In *Conceptual Modeling: Foundations and Applications*, Alexander T. Borgida, Vinay K. Chaudhri, Paolo Giorgini, and Eric S. Yu (Eds.). Lecture Notes In Computer Science, Vol. 5600. Springer-Verlag, Berlin, Heidelberg 99-121. DOI=10.1007/978-3-642-02463-4_7 http://dx.doi.org/10.1007/978-3-642-02463-4_7

[6] Andrew Monk and Steve Howard. 1998. *Methods & tools: the rich picture: a tool for reasoning about work context interactions* 5, 2 (March 1998), 21-30. DOI=10.1145/274430.274434 http://doi.acm.org/10.1145/274430.274434

[7] Eric Yu and John Mylopoulos. *"Why Goal-Oriented Requirements Engineering*". University of Toronto.