RummiTV: An Interactive Game for the Brazilian Digital TV System

Joel Santos, Erick Ratamero, João Paulo Arruda, Manoel Dantas,
Maria Luiza Sanchez, Débora C. Muchaluat Saade

***Resumo*—Este artigo apresenta o jogo eletrônico RummiTV, desenvolvido para o novo sistema brasileiro de televisão digital interativa. O jogo é uma adaptação do jogo Rummikub, um dos jogos de tabuleiro mais populares do mundo. Jogos proporcionam o desenvolvimento de crianças de maneira lúdica, contribuindo para iniciá-las e incentivá-las a utilizar o novo ambiente de TV digital interativa, ajudando a diminuir o problema da exclusão digital no país. O jogo RummiTV é uma aplicação híbrida para o ambiente de TV digital, pois foi desenvolvido utilizando a abordagem procedural através da linguagem Java, usando as APIs compatíveis com o GEM (Globally Executable Middleware), e também utilizando a abordagem declarativa através da linguagem NCL (Nested Context Language), adotada no middleware brasileiro Ginga.**

***Palavras-chave*—TV Digital, Aplicações Interativas, JavaTV, Ginga-J, Ginga-NCL, Jogos, Rummikub, RummiTV.**

***Abstract*— This work presents RummiTV, an electronic game developed as an interactive application for the new Brazilian digital TV standard. The game is an adaptation of Rummikub, one of the world's most popular board games. Games can help children's learning process in an entertaining manner, stimulating them to use the new digital TV interactive environment and contributing to reduce the digital exclusion problem in Brazil. RummiTV is a hybrid application for the digital TV platform, since it was developed using both procedural and declarative approaches. It is GEM-compatible (Globally Executable MHP), as the game is implemented is Java. Besides that, NCL (Nested Context Language), the declarative language adopted in the Brazilian middleware named Ginga, was also used in the development.**

***Keywords*— Digital TV, Interactive Applications, JavaTV, Ginga-J, Ginga-NCL, Games, Rummikub, RummiTV.**

I. Introduction

Recently the Brazilian government determined the new standard for the interactive digital television system. It is called ISDTV-T (International Standard for Digital Television Terrestrial) [15] and added innovations developed by researchers from Brazilian universities to the Japanese standard ISDB (Integrated Services Digital Broadcasting) [13]. With the new digital TV system, users can have access to interactive applications, using a return channel that allows data transmissions among users and application servers or among users directly. In this context, one of the biggest priorities of the Brazilian government is to reduce the problem of digital exclusion with the new TV system, allowing access to digital information to a great part of the population that today has no Internet access.

To encourage the use of the system, especially by children and young people, the development of educational games for this platform has a fundamental place. Games can help children's learning process in an entertaining manner, stimulating them to use the new digital TV interactive environment and contributing to reduce the digital exclusion problem in Brazil.

This work presents RummiTV, an educational electronic game developed for the new Brazilian interactive digital television standard. RummiTV is an adaptation of the Rummikub game [12] for the digital TV platform. Rummikub is one of the world's most famous board games designed for children over 8 years old. Besides the development of an interactive application for the new Brazilian TV system that can help reducing the digital exclusion problem in this country, another motivation for this work was the implementation of a free and open source electronic version of Rummikub that can also be used as a traditional TCP/IP application. One main concern in RummiTV was the design of a user interface that can be used in different platforms.

The development of applications for the digital TV environment can be done using procedural or declarative approaches. In this work, both were used, what makes RummiTV a hybrid application for the digital TV environment. For the development of the game, the procedural approach was used, which means, RummiTV was developed in Java. Besides the game itself, its instruction manual, including all documentation about how to play the game and user interface rules, was developed using the NCL language (Nested Context Language), also standardized in the Brazilian digital TV middleware named Ginga [3, 17, 21].

The remaining part of this paper is structured as follows. Section 2 gives a general view about the application development support in the Brazilian digital TV standard.

Joel Santos[1], Erick Ratamero[2], João Paulo Arruda[1], Manoel Dantas[1], Maria Luiza Sanchez[1], Débora C. Muchaluat Saade[1], [1,2]Laboratório MídiaCom, [2]Programa de Educação Tutorial (Grupo PET/Tele), Departamento de Engenharia de Telecomunicações, Universidade Federal Fluminense, R. Passo da Pátria, 156 – Bloco E – sala 421, São Domingos, Niterói, RJ, Brasil – 24210-240. E-mails: {joel, erick, joaoarruda, mdantas, mluiza, debora}@midiacom.uff.br.

Section 3 describes the RummiTV game. Section 4 comments the object-oriented modeling used in its implementation. Section 5 discusses some problems related to the Human-TV interface. Section 6 presents the instruction manual, developed as an NCL interactive multimedia document. Final remarks and future works are given in Section 7.

II. Application Development in the Brazilian Digital TV Standard

The Brazilian digital TV standard, like all others [8, 9, 13], allows the development of content using procedural and declarative approaches. The Brazilian middleware is called Ginga and supports the development of declarative applications, through the Ginga-NCL environment [16, 21], and also procedural ones, through the Ginga-J environment [15]. The Ginga standard offers innovations in relation to the previous digital TV standards and it is also compatible with the ITU international recommendations J.200 [4], J.201 [5] and J.202 [6]. Figure 1 illustrates the Ginga architecture [16, 21].

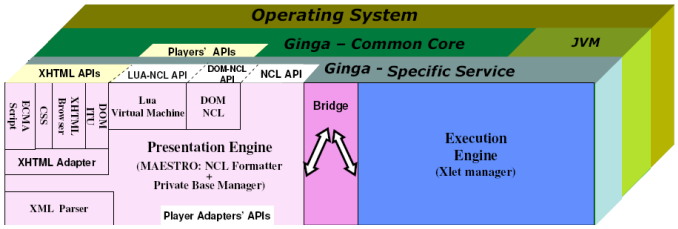

Fig. 1. Ginga architecture.

The declarative environment Ginga-NCL [16] adopts NCL (Nested Context Language), an XML-based language, for the creation of multimedia interactive content. NCL, in addition to other facilities [14], supports alternative content specification, simultaneous exhibition of contents in multiple outputs, integration of different kinds of media objects, including the ones defined by other declarative middleware standards, as BML (Broadcast Markup Language), DVB-HTML and ACAP-X, live edition, integration of objects with procedural content like Xlets (written in Java language) or NCLua (written in LUA language) [16].

The procedural environment Ginga-J [15] adopts the Java language, as a tool for the creation of interactive content, such as the American, European and Japanese digital TV standards do. The Ginga-J API (Application Programming Interface) is subdivided into three API groups called green, yellow and blue. The green API is compatible with the GEM (Globally Executable MHP) core [8], including JavaTV [18], DAVIC and HAVi, also available in the other digital TV standards. The yellow API includes a JMF (Java Media Framework) API, required for the development of advanced applications, like for example the ones that capture audio and video, among other facilities. The blue API offers facilities for integration of different input and output devices to the set-top box [9] and also provides the NCL Bridge, which allows the integration of procedural and declarative contents in the same application.

A wide literature about digital television and its different systems and standards can already be found today. On the other hand, the number of works that specifically discuss the development of interactive applications for this type of environment is still limited. Concerning digital TV systems, in a general way, there are related works focusing diverse aspects of this subject. For example, there are detailed descriptions of procedural and declarative standards [10], as well as discussions about the usability of applications in this new environment [19]. In some cases, there are implementations of procedural applications that interact with the television's traditional audiovisual content [11], bringing extra information about it or offering services like T-commerce or chats.

In the Brazilian case, as the standard was recently established, there are still a few documented works about interactive applications [2], especially about games.

III. RummiTV

Rummikub is an educative mathematical game, which aims at stimulating the reasoning capability of children and young people older than 8 years old. It can be played by 2, 3 or 4 people. Rummikub's characteristics were determinant for choosing it for the development of this project. Along with the educational benefits from the game itself, young users will also be in touch with the new learning and entertaining environment provided by the interactive digital TV system. RummiTV offers a simple graphical user interface, making playing the game a fun activity for young users.

RummiTV has a set of 104 pieces, divided into four colors, 26 green, 26 red, 26 blue, 26 black and also 2 (yellow) jokers. Each set of pieces of the same color has two sequences numbered from 1 to 13. Each player receives 14 pieces in the beginning of a match and those set of pieces stays in his (her) hand. The main objective of each player is to discard all pieces in the hand. In order to discard pieces, players can create compositions of pieces and place them at a shared board, called table, or they can modify, divide or gather the existing compositions inserting new pieces. Each composition at the shared board must have at least 3 pieces. Each composition can be a numerical sequence or a group. All pieces in a numerical sequence must have the same color. A group must have pieces with the same number but different colors. For completing numerical sequences or groups, players can use a joker. Figure 2 illustrates the RummiTV interface. The upper part of the figure shows the shared board containing numerical sequences and groups. The lower part of the figure shows the set of pieces in the player's hand.

The compositions placed at the shared board (table) can be seen and modified by all players. When it's his (her) time to play in a round, a player can make any modifications in the sequences or groups at the table in order to discard his (her)

pieces, but no piece can be taken from the shared board. A player may separate groups to create sequences and vice-versa, divide compositions to create others and substitute a joker for a piece and vice-versa. After a player modifies the table content and discards at least one of its pieces, all sequences and groups at the shared board must remain valid. If a player does not have any piece to discard or does not want to play, he (she) must buy a new piece from the deck that contains non-used pieces and wait until it's his (her) time to play again (next round). The player that firstly discards all pieces in his (her) hand is the winner of a RummiTV match.

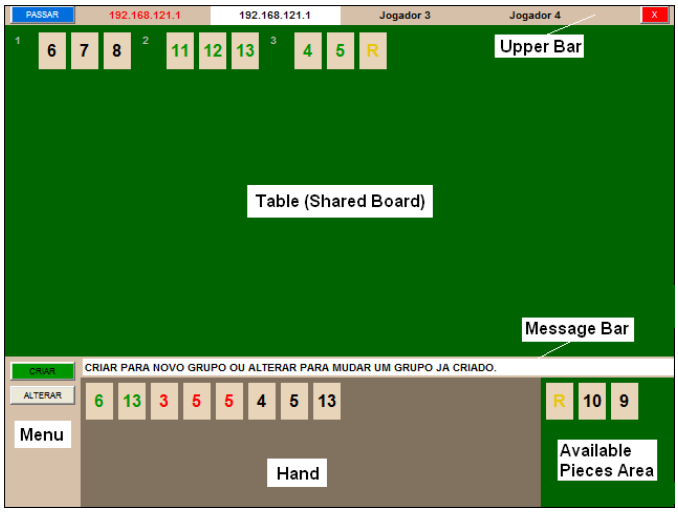

Fig. 2. RummiTV interface.

RummiTV was implemented using JavaTV. The reason for using a procedural approach in the development of RummiTV is that the game has specific rules and logic which demands a programming language to be developed. Therefore NCL (declarative language) could not be used for that. The Lua language (NCL's script language) could be used to develop the game, but at the time this project has begun, Lua had not been included in the Brazilian standard yet. Java is the only programming language compatible with GEM (Globally Executable MHP) – the ITU-T standard for digital TV – and supported in all standards [8], including the Brazilian system. Thus, the Java language was chosen for the development of RummiTV.

In order to make playing the game an easy task for young users, RummiTV also provides an instruction manual explaining the game rules and how to interact with its user interface. As the manual is an interactive multimedia document, it fits perfectly with the kind of content to be created with NCL. NCL is a declarative multimedia authoring language that provides support for the creation of synchronization relationships among different media, such as audio, video, image and text. Authoring multimedia documents with Java would be a lot more difficult.

As RummiTV was developed using both procedural and declarative approaches, it can be considered a hybrid application for the Brazilian digital TV platform. The game is GEM-compatible, so it runs on all digital TV systems, and its documentation can be easily ported to other systems declarative languages, if necessary.

Although RummiTV used both approaches in its development, the procedural and declarative parts are not fully integrated yet. This can be explained because of the lack of a full available implementation of the Ginga middleware, which includes the NCL Bridge. The procedural part itself uses the GEM-compatible Ginga-J green API, so the XLetView [20] environment, an emulator for the execution of XLets, was used to run the tests of this part. The instruction manual, developed with the NCL language, was tested with the Ginga-NCL virtual set-top box, available as free software [3].

IV. Object-Oriented Modeling and Implementation of RummiTV

RummiTV is an instantiation of GameTV, a framework that supports common characteristics of board games. GameTV models the dynamic of a game in a general manner and controls its players. Several classes are available in the GameTV framework, as it can be seen in Figure 3.

The GameManager class manages the existing games, makes Player registration in the system, controls the state of players (idle, waiting, ready, playing) and manages procedures for creating and ending a board or a match. The GameManager class verifies player states to handle message information in the system.

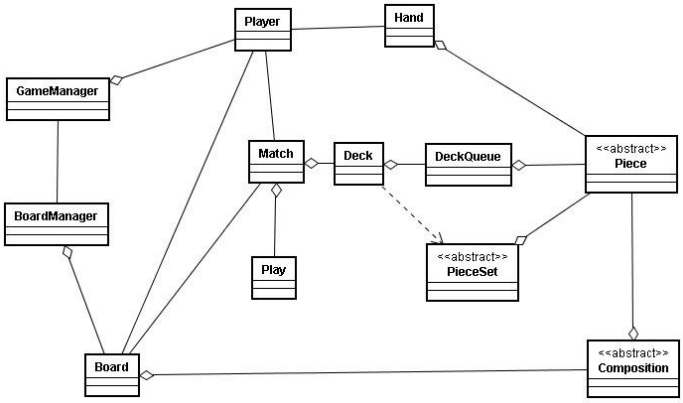

Fig. 3. GameTV framework.

The BoardManager class monitors a board, its associated players and its associated match. It controls the time to start a match, updates the list of players and removes them from a board. The Match class monitors a match, knows the player who has the right to play at a time and the winner of a match. It also performs procedures to end a match and controls rounds and the validation of a move.

The Deck class creates a set of PieceSet objects and controls procedures for shuffling decks, which are necessary to start a match. The DeckQueue class controls the queuing of pieces after being shuffled and delivers pieces to a player. The Play class performs procedures for verifying if a move made by a player is valid or invalid and restores the shared board to

its previous state if a move was not valid. Composition is an abstract class and can represent different compositions of pieces, like sequences and groups. Each Player has a Hand. The Hand class represents a set of pieces belonging to a player during a match. The PieceSet class includes the set of pieces of the same type.

RummiTV uses the GameTV framework and creates new classes according to specific game characteristics, as can be seen in Figure 4.

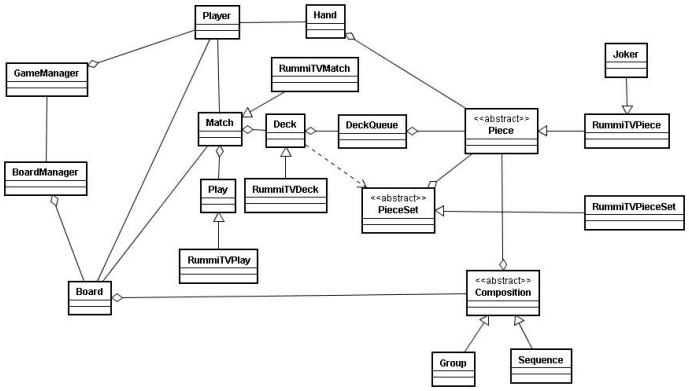

Fig. 4. RummiTV classes.

The RummiTVMatch class controls the RummiTV matches and is an extension of the Match Class from the GameTV framework. The RummiTVDeck class has four RummiTVPieceSet objects (one for each color) and two jokers (an extension of RummiTVPiece). The Board class has a set of Composition objects, which can be Sequences or Groups.

To facilitate programming and testing, an initial version of RummiTV was implemented as a stand-alone Java application. In parallel, test programs were implemented to better understand the environment and restrictions of a digital TV platform. Afterwards, a distributed client-server application was implemented using Java RMI (Remote Method Invocation) [7] to provide communication between client and server. RummiTV clients run on user set-top boxes and a central game server controls communication among them. As the game client runs on a user terminal, the multiplayer version is a distributed application.

This work is also concerned about performance, both in the application execution and in the use of the digital TV network subsystem avoiding intensive use of the return channel. Therefore, another implementation of GameTV and RummiTV has currently been developed using a TCP/IP application protocol based on sockets for communication between client and server, instead of Java RMI, avoiding RMI object serialization.

V. Human-TV Interface

Another concern in the development of this work was to create a user interface with great playability, once TV users have only a traditional remote control to interact, limiting their moves and access to menu options. Traditional computers have mouse pointing devices, which improve the flexibility of navigation on the screen, providing an easy way of moving pieces and selecting menu options. As RummiTV can be played in different environments, the same match may include one player connected by the TV set and another by a traditional computer. Thus, one challenge was to design an interface that can be used on different platforms that might not include a pointing device, such as a mouse.

The proposed solution works with the association of possible actions (showed on the screen by different button's names and colors) with the color keys and arrows in the digital TV remote control. Thus players have more moving capacity and easy access to menu options.

The RummiTV interface is illustrated in Figure 2 and is composed by six parts: the Upper Bar, the Table (shared board), the Message Bar, the Menu, The Hand and the Available Pieces area. The Upper Bar shows the names of the players and has buttons to end the round and to exit the game. The Table shows the shared board with all compositions created by the players. The Message Bar (below the shared board) shows, at each step, a message informing the player what to do and the forward steps. The Menu (lower left part) has buttons providing possible actions the player can take at each step. The Hand (lower center part) shows the player's pieces. Each player only sees its own hand. The Available Pieces area (lower right part) has the pieces the player withdrawn from the sequences and groups on the shared board and that must be relocated on the table during the same round.

As a use case of the graphical interface, suppose a player, in the digital TV environment using a remote control, who wishes to create a new composition (sequence or group). Figure 5 shows RummiTV running on the XLetView emulator.

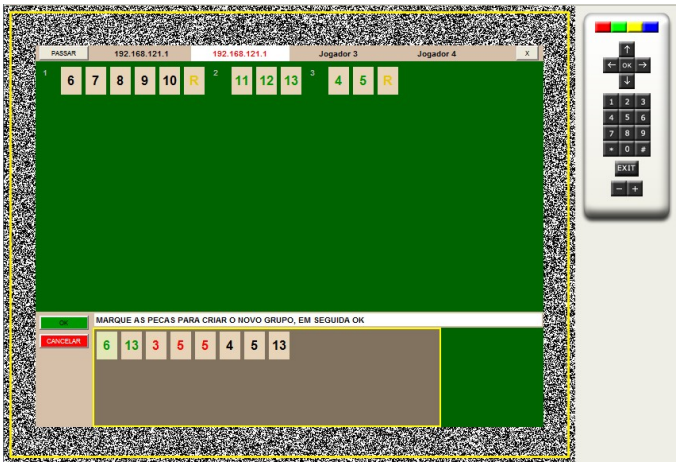

Fig. 5. RummiTV.

In the beginning of each round, the focus stays on the Menu with access to the buttons Create (*Criar*) and Change (*Alterar*), as already shown in Figure 2. By the association of the buttons on the screen with the color buttons of the remote

control, the user can choose the Create option by pressing the green button. Then, the screen focus goes to the Hand, where the user can choose pieces in order to create a new composition, as shown in Figure 5. Pieces can be chosen by the arrow keys and ok button of the remote control. To finish the move the player has to confirm his choice by pressing the green button of the remote control, which is associated to the Ok button in the graphical interface. If the player wants, he may cancel his choice by pressing the red button on the remote control, which is associated to the Cancel button in the graphical interface (see Figure 5). The screen focus can be changed by pressing the yellow button of the remote control, which works as the Tab key in a traditional keyboard. In Figure 5, the screen focus is on the Hand. That explains why the Hand has a yellow border. RummiTV interface is written in Portuguese for Brazilian users.

VI. RummiTV´s Instruction Manual

The RummiTV's instruction manual was developed using a declarative approach with the NCL language. The manual was logically structured in a way that it can be easily used for the documentation of other games. Its general structure is divided into two main parts. The first one informs the game rules and how to play it. The second one explains how to use the graphical interface, that is, how to make a move, select pieces and access menu options. Each part was defined as a distinct context node (<context>) in the main NCL document.

Each context node, representing one part of the manual, is composed by an audio object with the narration of the documentation synchronized with other context node components creating a sequential screen presentation. A partial structural view of the manual, created by the Composer software [3], can be seen in Figure 6.

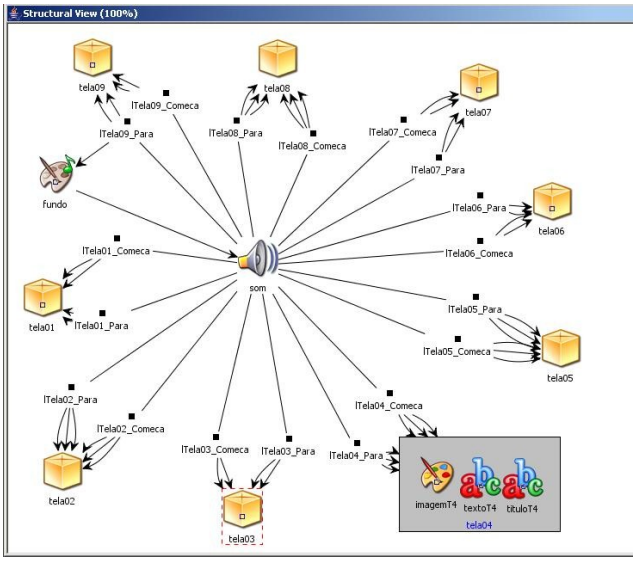

Fig. 6. Partial structural view of the manual.

Figure 6 shows a central audio object with several links to context nodes (represented as cubes in the figure). Each context node represents a manual screen, having a title, a text and an image or a video presented in parallel. A partial temporal view of the document is illustrated in Figure 7. The figure shows a background image (*fundo*) and a background audio (*som*), whose parts are synchronized with the corresponding screen (title, image and text) through NCL causal connectors, representing the Allen [1] temporal relations *starts* and *finishes*. When the specific audio part is played, the correspondent visual elements (title, image and text) are shown on the screen.

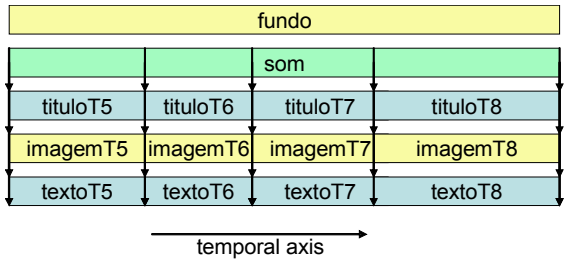

Fig. 7. Partial temporal view of the manual.

Figure 8 shows a screen example of the instruction manual explaining how to use jokers to create compositions (numerical sequences or groups).

Fig. 8. Screen example of the instruction manual.

VII. Final Remarks

Digital TV interactivity brings a huge number of services and utilities, including different ways of learning and entertainment, such as games. In the Brazilian scenario, where the government expects to reduce social exclusion with digital television, educational games play even a more important role. In this context, this work presented the RummiTV game, an adaptation of Rummikub, one of the most famous board games in the world, for the Brazilian digital TV platform.

The object-oriented distributed implementation of RummiTV in Java is already completed. Interface limitations found in a digital TV environment were considered in its

implementation, so the RummiTV client can run in platforms that provide a mouse or only a remote control. RummiTV allows players to use different environments, integrating TV users and personal computer users, since it provides a graphical interface designed for both platforms. As future work, we intend to run interface usability tests with children playing the game in a multi-user environment.

In order to facilitate the development of other games, a general framework for board games, called GameTV, was also presented in this work. Thus, as future work, we intend to develop other board games for digital TV environments using the GameTV framework.

The game's instruction manual was developed with NCL, the declarative language adopted by the Ginga middleware, included in the Brazilian digital TV standard. As it used both procedural and declarative paradigms in the application development, RummiTV can be considered a hybrid application. As soon as a full implementation of the Ginga middleware is available, the procedural and declarative parts will be integrated.

The Brazilian standard portable TV receiver specification does not include Ginga-J, but only Ginga-NCL. As a future work, we intend to develop a new RummiTV client using Lua, NCL's script language, instead of Java, so users would be able to connect and play with mobile phones and PDAs too.

References

[1] Allen J.F. Maintaining Knowledge about Temporal Intervals, Communications of the ACM, 26(11), 1983, 832-843.

[2] Andreata, Jomar; Montez, C. Uma aplicação para T-Ensino: TV-Chat. XI Brazilian Symposium on Multimedia Systems and the Web, Webmedia 2005, Poços de Caldas, 2005.

[3] Ginga, Brazilian's Digital TV Middleware, http://www.ginga.org.br, acessed in June 2008.

[4] ITU. ITU-T Recommendation J.200: Worldwide common core – Application Environment for Digital Interactive Television Series, 2001.

[5] ITU. ITU-T Recommendation J.201: Harmonization of Declarative Content Format for Interactive Television Applications, 2004.

[6] ITU. ITU-T Recommendation J.202: Harmonization of Procedural Content Formats for Interactive TV Applications, 2003.

[7] Java Remote Method Invocation – Java RMI. http://java.sun.com/javase/6/docs/technotes/guides/rmi/index.html, accessed in June 2008.

[8] Morris and Smith-Chaigneau, Interactive TV Standards - A Guide to MHP, OCAP and JavaTV, Focal Press, Elsevier, 2005.

[9] O´Driscoll, The Essential Guide to Digital Set-top Boxes and Interactive TV, Prentice Hall, 2000.

[10] Peng. "Digital Television Applications", Doctoral Dissertation, Helsinki University of Technology, Helsinki, Finland, November 2002.

[11] Peng and Vuorimaa. "Development of Java User Interface for Digital Television," Proceedings of the 8th WSCG International Conference on Computer Graphics, Visualization and Interactive Digital Media, February 2000, Czech Republic, pp. 120-125.

[12] Rummikub, http://www.rummikub.com, accessed in June 2008.

[13] Schwalb, ITV Handbook Technologies and Standards, IMSC Press, Prentice Hall, 2004.

[14] Silva, H.V.O., Muchaluat-Saade, D.C., Rodrigues, R.F., Soares, L.F.G. NCL 2.0: Integrating New Concepts to XML Modular Languages, ACM DocEng 2004, Wisconsin, 2004.

[15] Souza Filho, G.; Leite, L.E.; Batista, C.E. Ginga-J: The Procedural Middleware for the Brazilian Digital TV System. Journal of the Brazilian Computer Society, Vol. 12, num. 4, March 2007.

[16] Soares, L.F.G; Rodrigues, R.F.; Moreno, M.F. Ginga-NCL: The Declarative Environment of the Brazilian Digital TV System. Journal of the Brazilian Computer Society, Vol. 12, num. 4, March 2007.

[17] Soares, L.F.G., Souza Filho, G.L. Interactive Television in Brazil: System Software and the Digital Divide, EuroITV, 2007.

[18] Sun JavaTV - http://java.sun.com/products/javatv/, accessed in June 2008.

[19] Valdestilhas and Almeida. "A usabilidade no desenvolvimento de aplicações para TV Interativa." Brazilian Symposium on Computer Graphics and Image Processing, SIBGRAPI 2005, Natal, 2005.

[20] XletView Project Homepage - http://xletview.sourceforge.net/, accessed in June 2008.

[21] Norma ABNT 15606-2:2007, Televisão digital terrestre - Codificação de dados e especificações de transmissão para radiodifusão digital - Parte 2: Ginga-NCL para receptores fixos e móveis - Linguagem de aplicação XML para codificação de aplicações, 2007.