

# Improve Requirements Understanding by Playing Serious Games

David Gelperin  
ClearSpecs Enterprises  
2425 Zealand Ave. N.  
Golden Valley, MN 55427 USA  
Tel: +1 763-591-1534  
Email: [david@clearspecs.com](mailto:david@clearspecs.com)

Copyright © 2011 by ClearSpecs Enterprises.

**There are more things in heaven and earth, Horatio,  
than are dreamt of in your (requirements) philosophy.  
Hamlet act 1, scene 5**

**Abstract.** *Serious games* are games whose primary purpose is **not** entertainment. *Cooperative games* structure communication (e.g. contract bridge) and cooperation (e.g. soccer) between players in the same group.

Using serious games improves Requirements Understanding (RU). This paper defines six RU games, maps them into ‘initial understanding space’, and shows how to use a special Ouija board to choose the games to play on your project. An example of RU game selection is included.

## 1. Introduction

Here is a quiz.

In the context of a system to assess compliance of prescription drug ads to Federal Drug Administration guidelines, what is the meaning of the following requirement?

“Report each black box warning that is missing.”

Think for a minute or less. The answer is in Appendix A.

## 2. Serious, Cooperative Games

"What we've got here is a failure to communicate."  
– Cool Hand Luke (1967)

This is an unfortunate, but accurate assessment of many troubled projects. This paper is about using serious, cooperative games to improve communication and cooperation between customers and developers during requirements activities.

A *serious game* is one whose primary purpose is **not** entertainment. Parents of young children might use (serious) games to get toys picked up or teeth brushed. Serious games have also been used for training of skills and strategic thinking [Card 1985], coaching intercultural communication [Lane et. al. 2008], supporting Agile development [Hohmann 2006], as well as

physical rehabilitation [Conconi et. al. 2008] and psychological therapy [Gamberini et. al. 2008]. The concept has been around for at least 40 years [Abt 1970].

There are at least two ways that serious games can be used in requirements development. One use is to help customers identify and prioritize their requirements. Examples of these games are: Product Box, Speed Boat, Buy a Feature, and 20/20 Vision [Hohmann 2006]. The other use is to help customers and developers communicate and cooperate. That is the focus of this paper.

A game is *cooperative* if two or more players must work together to achieve its goals. Cooperative games structure the communication (e.g. contract bridge) and cooperation (e.g. soccer) between players on the same team. Some serious, cooperative games have the power to structure interactions between customers and developers to support the goals of requirements development.

### 3. Understanding

The goal of requirements development is to help developers acquire a sufficiently deep understanding of customer and user needs, via effective communication and cooperation, and the feasibility of meeting those needs, so they can do their jobs. If the developers already have much of this understanding, little requirements development is needed.

#### 3.1. Levels of Understanding

3.1.1. **No Understanding** If developers or customers have little understanding of the application domain, then there will be no understanding or worse, misunderstanding of the requirements. For example, consider the requirement: Display atelectasis findings with highlighting. Looking in a medical dictionary we find that “atelectasis” means ‘collapse of all or part of a lung’. Unfortunately no dictionary is sufficient for complete understanding of an application domain.

3.1.2. **Superficial Understanding** Requirements may be specified with familiar words used in unfamiliar ways. For example, a requirement might be : “ Report each black box warning that is missing.” While the words are familiar, understanding its detailed meaning requires an understanding of the language used to describe prescription drug ad compliance with FDA guidelines. The meaning of this requirement is discussed in Section 4.5.2.

3.1.3. **Limited Understanding** This entails an understanding of some, but not all, of the fundamental entities, activities, relationships, and consequences in the application domain. For example, the investment rating companies had a limited understanding of mortgage backed securities prior to the recent financial crisis e.g. they didn’t know enough about consequences. They based their ratings on their understanding, but forgot to tell us that it was limited.

3.1.4. **Deep Understanding** This entails an understanding of almost all of the fundamental entities, activities, relationships, and consequences in the application domain. This does not entail knowing everything, but knowing almost all important things.

3.1.5. **Relatively Complete Understanding** This entails a deep understanding of all requirements needed to perform.

### 3.2. Initial Understanding

Communication about requirements should be a function of who understands the requirements and at what level.

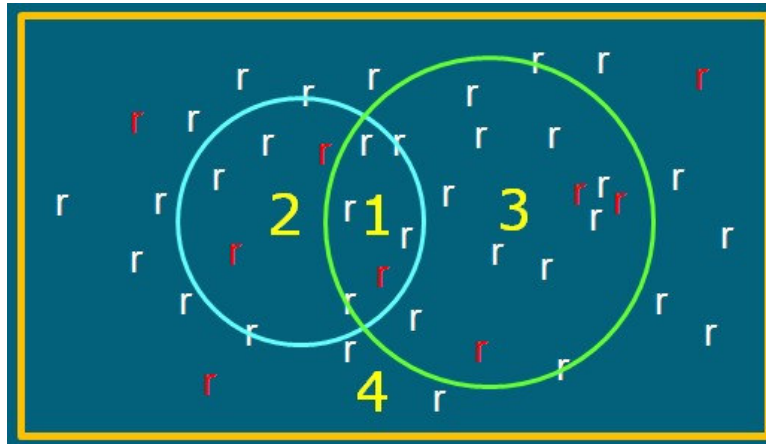


Figure 1. Deep Understanding of Requirements at Project Beginning

The letters in the rectangle of Figure 1 represent the actual requirements on your next project. The red letters are your mission-critical requirements. The letters partially or fully inside the blue circle are the requirements deeply understood by the developers at the beginning of the project. The proportion of requirements understood by the developers (i.e. inside the blue circle) at the beginning of a project can vary widely from Very Few to Most. The letters inside the green circle are the requirements deeply understood by customers at the beginning of the project. This creates four areas : Area 1 contains the mutually understood requirements, Area 2 contains the requirements understood by the developers only, Area 3 contains the requirements understood by the customers only, and Area 4 contains the requirements deeply understood by neither party.

Most requirements development approaches e.g. win-win spiral or agile, assume that all activity is in Area 3 and that the developers understand the application domain. Effective communication and cooperation should differ for requirements in these four different areas.

## 4. Six RU games

We illustrate different approaches to communication and cooperation in the four understanding areas by describing six RU games :

- Bridge Bidding Conversations, for mutually understood requirements
- 10/20 Questions, for requirements deeply understood by developers only
- Jigsaw Puzzling, for requirements deeply understood by customers only
- Scavenger Hunt 1/2, for requirements deeply understood by neither party
- Enculturation, for use when developers do not understand the application domain and
- Decoding, for use when developers do not understand an existing system that must be changed.

Each game has the same goal:

to help developers acquire a sufficiently deep understanding of customer and user needs, via effective communication and cooperation, and the feasibility of meeting those needs, so they can do their jobs

## **4.1. Bridge Bidding Conversations**

**4.1.1. Contract Bridge** Contract bridge is a four-handed card game using a standard 52-card deck to create 13-card hands in a series of deals that each begins with a bidding phase followed by a trick-taking phase. Bridge is played by two pairs of cooperative partners who sit opposite each other. The goal is to score as many points as possible in each deal. Sometimes this means bidding and making a contract for a specific number of tricks (set of 4 cards) and sometimes it means stopping the opponents from making their contract.

An important aspect of the game is the bidding phase. During bidding, one or both partnerships make legal bids in an effort (1) to determine a makeable contract with the greatest number of points or (2) to disrupt the opponents bidding or deny them the final contract. A bid is a commitment to take the number of tricks bid plus six more e.g. 3 spades is a commitment to take at least 9 tricks.

Each partnership uses their own bidding system which is a collection of agreements and conventions for describing the meaning of legal bids e.g. the bid of 1 spade in first position describes a specific set of hands. The meaning of a bid (e.g. 1 spade) can change based on opening position (e.g. 1<sup>st</sup> vs. 3<sup>rd</sup>) and other factors (e.g. vulnerability). A bidding convention assigns specific meanings, sometimes “unnatural”, to the legal bids. For example, bidding 4 clubs in some situations does not describe the bidder’s hand, but asks the partner to tell how many aces they hold using a coded set of legal responses i.e., a 4 heart response means one ace. Thus, a bidding system is a codified, context-sensitive language that allows partners to exchange information about their card holdings.

**4.1.2 Bidding Conversations** When customers have bridge bidding conversations with developers, they use a codified, context-sensitive language to exchange information about system requirements. Both parties must have a deep understanding of the application domain (e.g. security) and the system to be developed or modified i.e. be in Area 1.

Two parties naturally use bridge bidding conversations when they have significant shared experience e.g. long married couples. Such sparse exchanges communicate effectively because of shared experience and understanding. The strategy is to use a little (language) to communicate a lot (of information). A third party who does not “know” the language or does not have the appropriate experience will not understand the words, phrases, and acronyms in the exchange, even though they might think they do.

Consider the following bridge bidding conversation in the domain of flight control systems (FCS):

Product Manager: Z-3 FCS is based on Y-15  
[Meaning depends on knowledge of the Y-15 FCS  
and interpretation of “is based on”]

Developer: Major changes?

Product Manager: Tighter controls, ...  
[“Tighter” may be an example of *intentional imprecision*  
– defined in 4.1.3. below]

**4.1.3 Bidding Conventions** Bridge bidding conversations may use conventions e.g. *intentional imprecision*. Using intentional imprecision invokes a social contract between the customer, who provides an explicit description of need (i.e., vague requirement) and the developers. Developers are expected to identify alternative implementations and the cost of each. Alternatives are presented to the customer who selects the precise meaning of the requirement.

Consider the following example. The product manager for a new blood analyzer specifies that average analysis cycle time must be *reduced*. The current cycle time = 3.5 minutes and the competitor best = 3.1 minutes.

The developers perform research and respond that average cycle time can be reduced as follows:

Table 1: Summary of alternative meanings for “reduced”

Reduced cycle time	Strategy	Increased cost per unit	Increased development time
3.2 min.	Modify analysis algorithm	\$0	4 weeks
2.8 min.	Faster standard processor	\$150	6 weeks
2.5 min.	Faster custom processor	\$400	16 weeks

Based on a set of marketing assumptions, the product manager chooses the optimal cycle time with its associated cost and time increases.

## 4.2. Ten or Twenty Questions

**4.2.1. Twenty Questions** Traditional 20 Questions is a spoken game where a player is chosen to be the *answerer* and that person chooses a subject e.g., *Gone With The Wind*. The other players, the questioners, must discover the subject. Each questioner in turn asks a question to narrow the possibilities e.g. Is the subject a person? The questioners win if they guess the subject using 20 questions or fewer, otherwise the answerer wins. Questioners strive to guess the subject using the fewest questions.

**4.2.2. 10/20 Questions** This is a variant of 20 Questions where developers, who have a significant understanding of the type of system that a customer wants, try to discover a customer’s detailed requirements. Developers may play 10 Questions when they only have a

moderate understanding of the needed system or when there are only a few pieces of information that a customer must supply. 10/20 Questions is played in Area 2.

In the context of ecommerce website development by an experienced team, a novice customer might be asked:

Of the 5 websites we showed you:

- a. which 2 are most appealing and why?
- b. which 2 are least appealing and why?

### ***4.3. Jigsaw Puzzling***

This game is played in Area 3, when developers understand the application domain. The developers assemble a picture out of a collection of irregular shaped pieces by fitting them together. The customers supply the pieces. Some pieces result from model building e.g. use cases, state tables, or acceptance tests.

The clarity of the picture depends on the quality of the pieces. The speed of assembly depends on the sequence in which the pieces are supplied and the recognition by the developers of links between pieces. This is the game most people think of when they think about requirements development.

### ***4.4. Scavenger Hunt 1 and 2***

4.4.1. **Scavenger Hunt 1** A traditional scavenger hunt is a game in which teams of players (customers and developers) try to gather all the items (requirements) on a list provided by the organizers. Played when a source of deep understanding is available.

4.4.2. **Scavenger Hunt 2** Another version of scavenger hunt has teams (customers and developers) solve riddles and follow clues to items (requirements) and other riddles. Played when no source of deep understanding is available. Expect many mistakes.

These games are played in Area 4. The tactics listed in 4.5.1. may be useful.

### ***4.5. Enculturation***

4.5.1. **Enculturation** Enculturation is a preparation process (game) by which a person (developer) learns to understand a culture and acquires values and behaviors necessary or suitable in that culture. Successful enculturation results in competence in the language (nouns, verbs, and adjectives), values and rituals of the culture. This characterization is derived from Wikipedia definitions. Enculturation is played in Areas 3 and 4, when developers are unfamiliar with the application domain.

An enculturating learner can use any of the following tactics:

- Study similar systems
- Read articles and documents
- Watch activities
- Join in activities
- Develop models
  - stories
  - use cases
  - test cases
  - prototypes
- Interact with practitioners
- Collaborate with mentors

4.5.2. **Enculturation Example** In the context of a system to assess compliance of prescription drug ads to FDA guidelines, successful enculturation results in an understanding of:

“Report each black box warning that is missing.”

While you are familiar with each of the words in this sentence, you do not understand its meaning i.e., you have a superficial understanding. This is because you are unfamiliar with the domain e.g., the rules for prescription drug ad compliance, and the meaning of these familiar words in this domain. We will focus on the meaning of: “black box warning”, “each”, and “missing”.

4.5.2.1. **black box warnings** All prescription drugs come with a Federal Drug Administration (FDA) approved *package insert* containing information about the drug. Among this information are “indications” – medical conditions the drug is approved to treat, “contraindications” – medical conditions in which use of the drug is absolutely or relatively inadvisable, and “black box warnings” – medical conditions e.g. hypertension (high blood pressure), in which the drug should only be used in extreme circumstances because the drug carries a significant risk of serious or even life-threatening adverse effects.

4.5.2.2. **each black box warning** The FDA requires that a *summary of patient risk information* from the package insert appear at least once in every prescription drug ad. Each risk summary must contain all the black box warnings i.e., all medical conditions from the black box warning section of the package insert. The FDA requires that a risk summary appear on every page or every two facing pages of a medical brochure. This means that a six-page brochure requires at least two risk summaries. Thus “each” black box warning could refer to (1) each of the conditions in the black box warning section of the package insert or (2) each occurrence of one of the conditions in each of the risk summaries or (3) both. In fact, it refers to both. Without context, the ambiguity of “each” is not obvious.

4.5.2.3. **each black box warning that is missing** A black box warning e.g., medical condition such as hypertension, could appear in none of or just one of multiple risk summaries. However, what if the black box warning “hypertension” appears in none of the risk summaries, but “high blood pressure” appears in each one? The FDA allows, and sometimes encourages, the use of “familiar phrases” in place of medical terms in prescription drug ads.

The FDA also allows the use of hypernyms i.e., general categories, for medical terms e.g. “blood pressure problems” for hypertension when they appear in warnings about risk. When valid synonyms and hypernyms are used correctly, “hypertension” is not missing.

What about using hyponyms i.e., instances or proper subsets? For example, using “severe hypertension” rather than “hypertension” alone in a warning about risk. The FDA does not allow the approved scope of patient risk to be reduced, so hyponyms may not be used. This means that if the approved black box warning is “hypertension” and the ad only contains “severe hypertension”, then the approved black box warning “hypertension” is “missing”.

4.5.2.4. **Summary** To understand the requirement:

“Report each black box warning that is missing.”

you must understand FDA rules for : (1) information in a package insert, (2) information in a package insert that must be in a risk summary, (3) number of times a risk summary must appear in a prescription drug ad, and (4) use of synonyms, hypernyms, and hyponyms.

Understanding these concepts increases your competence in the language (nouns and adjectives) of the application.

4.5.3. **Coached Enculturation** Coached Enculturation means that one or more people guide the novice by providing information and direction. If you followed the example in 4.5.2, you experienced coached enculturation. Coaching requires someone with sufficiently deep understanding of the culture and willingness to coach.

A coach can use any of the following tactics to guide developers:

- Identify similar systems and guide interaction
- Identify readings
- Create presentations or documents
- Answer questions
- Create a domain glossary
- Collaborate in model development
- Organize requirements understanding reviews
- Select activities for observation and participation
- Collaborate in observation and participation
- Select helpful practitioners

4.5.4. **Curse of Knowledge** Some customers may find it difficult to provide guidance because they are too experienced i.e., know too much. This has been called “The Curse of Knowledge” [Heath 2007].

“Novices perceive concrete experiences as concrete experiences. Experts perceive concrete experiences as concrete examples of patterns and insights learned through years of experience.

Because experts are capable of seeing these patterns, they naturally want to talk about patterns rather than concrete experiences e.g., they want to talk about chess strategy rather than bishops moving diagonally.”

This preference for abstraction means that some experienced customers can't provide guidance or details i.e., deepen developer understanding, because they are unwilling or unable to discuss concrete details.

## **4.6. Decoding**

This game is played in Areas 3 and 4, when system changes are required, but developers are unfamiliar with the history or internal organization of the system. Exploration is a serious, but not cooperative, game of exploring and mapping to determine if required changes are feasible. Decoding may be coached, if a developer, not on the project but with deep understanding of the system, is available to provide guidance.

## **5. Selecting appropriate games**

Having described six RU games (Bridge Bidding Conversations, 10/20 Questions, Jigsaw Puzzling, Scavenger Hunt 1/2, Enculturation, and Decoding), the problem now is to decide which to play on your projects. The decision process has 3 steps:

1. Determine customer and user needs
2. Envision a system with capabilities and features
3. Use the RU Ouija Board (described in 5.1.2 below) to choose appropriate games for discovering and communicating requirements

### **5.1 Using the RU Ouija board**

#### **5.1.1. The Classic Ouija Board**



Figure 2. Classic Ouija Board

A Ouija board is a flat board marked with letters, numbers, and other symbols, supposedly used to communicate with spirits during a séance. A séance participant asks a question and the Ouija board uses a planchette (small heart-shaped piece of wood) or movable indicator to provide the spirit's answer. The fingers of the questioning participant are placed on the planchette, which then moves about the board to spell out the answer [based on Wikipedia definitions].

### 5.1.2. The RU Ouija Board

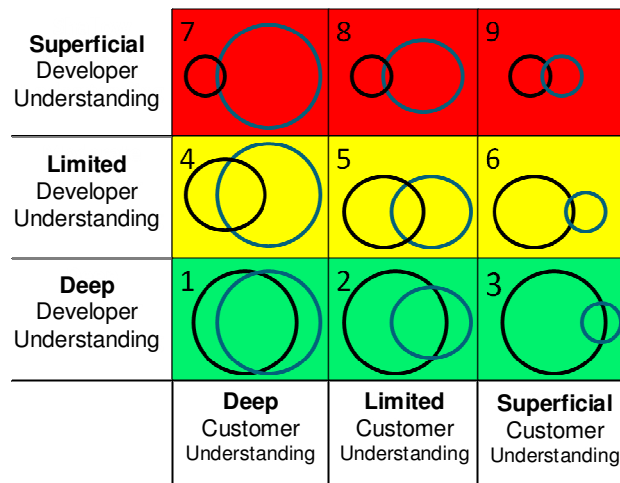


Figure 3. RU Ouija Board

The RU Ouija board is a grid of cells that each correspond to the initial level of developer deep understanding and the initial level of customer deep understanding. Unlike the classic Ouija board, mapping a system element (ranging from a whole system to a portion of a feature) to a cell on the RU board provides a context-sensitive answer to the single question: “Which RU games should we play to develop and communicate the requirements for this element? Rather than messages from spirits, the answers are derived from the following table.

Table 2. RU Ouija Answer Sheet

<b>Cell</b>	<b>D U</b>	<b>C U</b>	<b>Primary Games</b>
1	D	D	<i>BB Conversations (with conventions)</i>
2	D	L	<i>20 Questions, BB Conversations</i>
3	D	S	<i>20 Questions</i>
4	L	D	<i>10 Questions, BB Conversations, Jigsaw Puzzling, Coached Enculturation, Decoding</i>
5	L	L	<i>10 Questions, BB Conversations, Jigsaw Puzzling, Scavenger Hunt 1/2, Coached Enculturation, Decoding</i>
6	L	S	<i>10 Questions, Scavenger Hunt 1/2, Decoding</i>
7	S	D	<i>Jigsaw Puzzling, Coached Enculturation, Decoding</i>
8	S	L	<i>Jigsaw Puzzling, Scavenger Hunt 1/2, Coached Enculturation, Decoding</i>
9	S	S	<i>Scavenger Hunt 1/2, Decoding</i>

## 5.2. Example of Games Selection

Let's consider a realistic example of games selection. In the domain of ecommerce website development, we find our cooperating players, Naomi, the customer and redPear, the developers.



Figure 3. Naomi, the customer



Figure 4. redPear, the developers

Naomi wants to create Kidsideals, a deals website for kids and families that is like Groupon, but with important differences. To determine the games that Naomi and redPear should play to develop and communicate the Kidsideals requirements, their initial levels of understanding must be assessed.

redPear's initial understanding is limited since they have experience developing ecommerce websites, but no knowledge of deal websites, Groupon, or the Kidsideals differences. Naomi's initial understanding is deep since she has experience using ecommerce websites (e.g. Amazon), experience with Groupon as a seller and a buyer, and understanding of the Kidsideals differences. This causes the Kidsideals development project to be mapped into cell 4.

The answer sheet suggests 5 games: 10 Questions, Bridge Bidding Conversations, Jigsaw Puzzling, Coached Enculturation, and Exploration (not needed because this is new development). 10 Questions will be directed by redPear on the portions of Kidsideals that are not unique to deal sites. Naomi and redPear will have Bridge Bidding conversations about these same portions. Naomi will coach the enculturation of redPear on deal sites and Groupon, and then use Jigsaw Puzzling to communicate the Kidsideals differences. She chooses guiding interaction of redPear with Groupon and one or two other deal sites and answering questions as her primary guide tactics from the list in 4.5.3.

## 5.3. System mapping possibilities

There is a spectrum of possibilities when mapping system elements onto the RU Ouija board. The entire system may map to a single cell, various capabilities or features may map to different cells, or partial capabilities or features may map to different cells. The more complex

the mapping, the greater its value because **the mapping enables customers and developers to understand the diverse forms and occurrences of cooperation needed to be successful in discovering and communicating requirements.** You can see this in the example in 5.2.

## 6. Conclusion

Playing serious games to improve requirements development entails mapping the initial understanding of capabilities and features onto the RU Ouija board. Accurate mapping requires a culture of open and safe communication that encourages truth telling. The RU Ouija answer sheet suggests the games needed for each system element mapped into each cell. Customers and developers need to understand how to play each serious RU game effectively.

Experimentation, mentoring, and retrospectives may improve play. Help from an experienced developer or subject matter expert may change the games (i.e. cell).

Using cooperative RU games has been described in the context of customer-developer communication. These games are equally useful in customer-customer and developer-developer communication about requirements. The usage goal is still to enable information receivers to do their jobs. Note that the games used during a project for customer-developer communication may not be the same as those used in the other two domains.

Almost all work in requirements development addresses aspects of a single game – jigsaw puzzling. We have shown that there are four areas of initial understanding. This suggests that projects should use more than one game. If you only play one game and it doesn't match your communication needs, the results can be very very expensive. Worse, you can't learn from failure when root causes are not understood.

Games have goals, roles, rules, and strategies. Using a gaming framework allows project members to understand and focus on their communication and cooperation responsibilities. Playing appropriate games improves project outcomes.

Games are serious business.

## Appendix A

The correct answer to “What does this requirement mean?” is “I don't know.” Other answers are not only wrong, but dangerous.

The meaning of this requirement is discussed in Section 4.5.2.

## References

1. Abt, Clark C. 1970 *Serious Games* Viking Press
2. Card, Orson Scott 1985 *Ender's Game* Tor Science Fiction
3. Cockburn, Alistair 2006 *Agile Software Development: The Cooperative Game* Addison-Wesley Professional
4. Conconi, Alex Todor Ganchev, Otilia Kocsis, George Papadopoulos, Fernando Fernandez-Aranda, And Susana Jimenez-Murcia 2008 "Playmancer: A Serious Gaming 3D Environment" In Proceedings of The International Conference On Automated Solutions For Cross Media Content And Multi-Channel Distribution 111-117 IEEE Computer Society
5. Gamberini, Luciano Giacinto Barresi, Alice Majer, And Fabiola Scarpetta 2008 "A Game A Day Keeps The Doctor Away: A Short Review Of Computer Games In Mental Healthcare" Journal Of Cybertherapy & Rehabilitation 1 (2): 127-145 Virtual Reality Medical Institute
6. Heath, Chip And Dan Heath 2007 *Made To Stick: Why Some Ideas Survive and Others Die* Random House
7. Hohmann, Luke 2006 *Innovation Games: Creating Breakthrough Products Through Collaborative Play* Addison-Wesley Professional
8. Lane, H. Chad Matthew Hays, Mark Core, Dave Gomboc, Eric Forbell, and Milton Rosenberg 2008 "Coaching Intercultural Communication in a Serious Game" In Proceedings of the 16th International Conference on Computers in Education, 35-42 Asia-Pacific Society for Computers in Education
9. McGonigal, Jane 2011 *Reality is Broken : Why Games Make Us Better and How They Can Change the World* Penguin Press

## Biography

David is Chief Technology Officer of ClearSpecs Enterprises. He has more than 40 years experience in software engineering with an emphasis on requirements risk management as well as software quality, verification, and test. David cofounded Software Quality Engineering and catalyzed the launch of Better Software magazine. More information is available at [www.clearspecs.com](http://www.clearspecs.com) (under the About tab).