Título Notas de proceso de Grounded Theory

Tipo de Producto Informe técnico

Autores Oliveros, Alejandro y Cuadrado, Fernanda

Código del Proyecto y Título del Proyecto

P15T01 - Ingeniería de requerimientos para aplicaciones Web. Fundamentos, requerimientos no funcionales y stakeholders

Responsable del Proyecto

Oliveros, Alejandro

Línea

Ingeniería de Requerimientos - REQ

Área Temática

Tecnología de la Información y de la Comunicación - TIC

Fecha

Abril 2016

INTEC
Instituto de Tecnología

UADE
FUNDACIÓN UADE

Notas de proceso de *Grounded Theory*

Versión 1.0

Proyecto “Ingeniería de Requerimientos de Aplicaciones Web”

INTEC – Informe Técnico

Facultad de Ciencias e Ingeniería

Universidad Argentina de la Empresa

Abril de 2016

Alejandro Oliveros
Fernanda Cuadrado

Contenido

Resumen

Este documento presenta los elementos básicos de Grounded Theory (GT) utilizados en el proyecto de investigación de las prácticas de Ingeniería de Requerimientos utilizadas en el desarrollo de aplicaciones Web. GT es un enfoque experimental cualitativo para generar teorías que expliquen un corpus de datos. El enfoque principal de GT es desarrollar un proceso mediante el cual la teoría “emerge” de los datos. El concepto de que la teoría “emerge” de los datos tiene un significado muy preciso en este enfoque. La teoría que se obtiene provee una explicación de los datos. GT ya ha sido aplicada con éxito en la Ingeniería de Software y en la Ingeniería de Requerimientos en particular

Abstract

This document presents the basic elements of Grounded Theory (GT) used in the research project of Requirements Engineering practices used in the development of Web applications. GT is a qualitative experimental approach to generate theories that explain a corpus of data. The main focus of GT is to develop a process by which the theory "emerges" from the data. The concept that the theory "emerges" data has a very precise meaning in this approach. The theory provides an explanation of obtained data. GT has already been applied successfully in Software Engineering and Requirements Engineering in particular.

1 Propósito de Grounded Theory

GT es la generación sistemática de teoría a partir de datos analizados mediante un método riguroso de investigación. El énfasis lo pone en el descubrimiento de las principales preocupaciones de la mayoría de participantes y en la generación de una teoría para explicar cómo ellos ven a resolviendo esta preocupación central. (“The emphasis is on uncovering the main concerns of a

majority of participants and generation of a theory to explain how they go about resolving this main concern") [1].

Ejemplo. La preocupación principal puede asegurar el involucramiento del usuario en el desarrollo de software puede ser una preocupación de un equipo de desarrollo. El investigador debe descubrir esta preocupación principal y presentar la teoría de cómo se puede resolver.

El propósito de la GT es descubrir el problema principal en un área sustantiva y la solución de ese problema [2].

2 Proceso de codificación abierta

Básicamente se sigue el proceso según los lineamientos que planteó Hoda como resumen [1], al que se enriquece con otros puntos de vista pero tratando de seguir ese modelo. Los datos se codifican línea por línea y son comparados entre sí buscando similitudes y diferencias [2]

2.1 Esquema de proceso de open coding

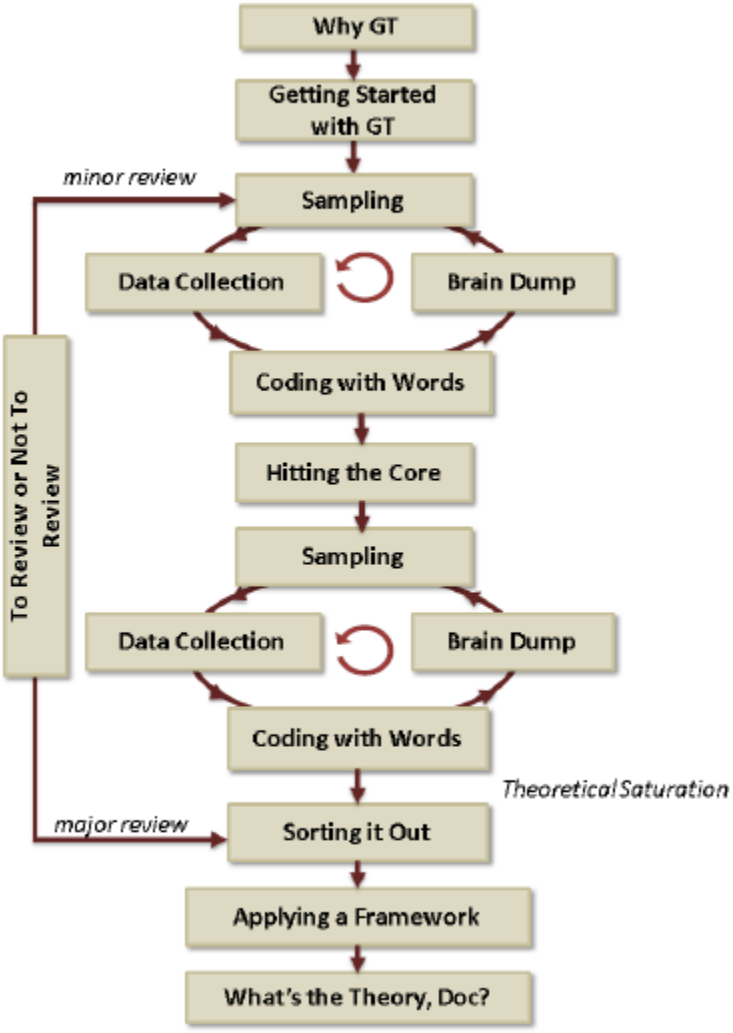


Fig. 1. Patrones de GT

Resumiendo los puntos que plantea Hoda, se tienen los siguientes pasos:

1. Marcar *puntos clave* en cada documento, esto se refiere a aquellas frases que se consideran relevantes para ser consideradas
2. Un *código* es una frase de 2 o 3 palabras que se asigna a cada punto clave. En la asignación de códigos, debe tenerse en cuenta:

(a) El investigador debe mantener la mente libre de códigos preconcebidos, de allí lo de *open*

(b) Los *códigos* introducidos se comparan con los restantes del documento y los de otros documentos este es el *método de comparación constante* (GT's constant comparison method-CCM)

3. Los *códigos* se agrupan en *conceptos* (= unidades de mayor nivel de abstracción)

   (a) Los *conceptos* introducidos se comparan con los restantes del documento y los otros documentos (aplicando de nuevo el CCM)

   (b) Este agrupamiento de conceptos no se hace después de la asignación de códigos, se realiza al mismo tiempo que la codificación

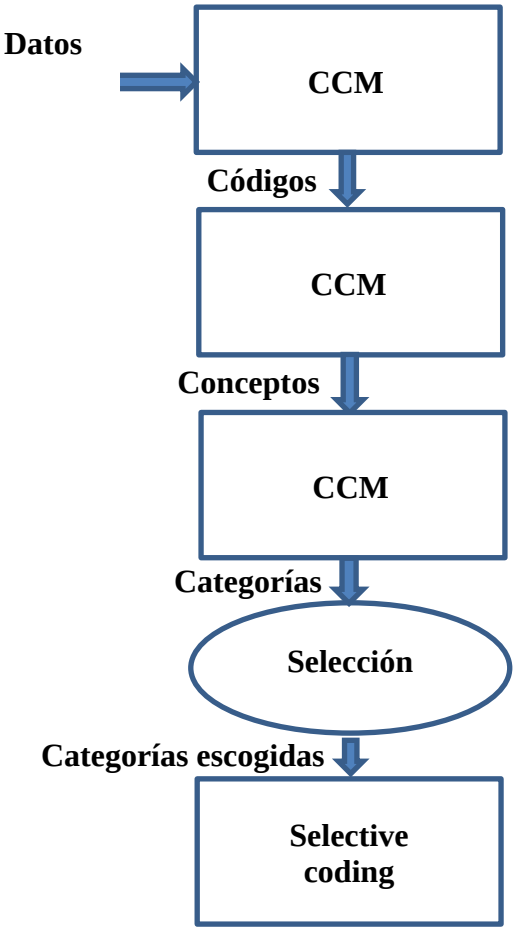
4. Nuevamente se aplica el CMM produciendo en tercer nivel de abstracción, las *categorías* mediante el agrupamiento de los conceptos.
5. Codificación selectiva (*Selective coding*). Se reitera CCM pero aplicado a un conjunto limitado de categorías: la categoría *core* y las estrechamente relacionadas con ella [¿cuáles?, ¿cómo se seleccionan?, ¿cómo detectamos que emerge la *core category*]

Hay dos codificaciones [2]: *codificación sustantiva* y *codificación teórica*. La codificación teórica incluye encontrar los códigos teóricos que integraran la teoría sustantiva.

Los códigos de categorías tienen un significado muy preciso: "A GT study's theoretical code is the relational model through which all substantive codes/categories are related to the core category."[2]

Las dos codificaciones se hacen en paralelo, se realimentan una a la otra. Hay apenas una diferencia de énfasis, Glaser: el investigador "will focus relatively more on substantive coding when

discovering codes within the data, and more on theoretical coding when theoretically sorting and integrating his memos" [2].



2.2 Texto de la cita original de Hoda

“Open coding begins by collating key points from every interview transcript. Then a code – a phrase summarizing the key point in 2 or 3 words – is assigned to each key point.

“The researcher should try to keep their minds free of ay preconceived ideas while coding – hence the term 'open' coding. The codes arising out of each interview are constantly compared against the codes from the same interview, and those from other interviews and observations. This is GT’s constant comparison method.

“The constant comparison method is used again to group these codes to produce units of a higher level of abstraction, called concepts in GT. Finally the constant comparison method is repeated on concepts to produce a third level of abstraction called categories

“Once the main theme of the research is discovered (explained in Pattern #: hitting the core) the researcher can move into Selective coding. Selective coding is the same coding procedures but applied on a limited set of categories – the core category and those closely related to the core. Selective coding typically progresses faster due to a constrained focus.”

Fuente: [1]

2.3 Memoing

Es el proceso de escribir el conocimiento que el investigador va acumulando a medida que entrevista, observa y analiza los datos.

Permite al investigador volcar ideas, percepciones, pensamientos y nuevas preguntas en notas explícitas. Permiten explorar las relaciones entre conceptos y categorías. Posteriormente son útiles para construir teoría [1].

2.4 Informe de la teoría

- Los resultados siguen el esquema generado como resultado del sorting
- La teoría resultante se resume en una pocas sentencias
- También puede representarse en forma diagramática
- Usar citas de los participantes y textuales ayuda a fundar la teoría

3 Codificación teórica [1]

Glaser introdujo el concepto de familias de códigos teóricos que los agrupan. Los códigos y las familias no son mutuamente excluyentes.

Los códigos teóricos no deben emerger de los datos, no de los preconceptos del investigador. Los códigos deben emerger, no provenir de conjeturas. Sin los códigos teóricos, los códigos sustantivos son meras descripciones y estas son preocupación de los métodos cualitativos de investigación (como fenomenología o etnografía). Ayudas para la codificación teórica:

1. *Theoretical sensitivity*. Conocer las familias de códigos teóricos ayuda, así como otras investigaciones
2. *In vivo codes*. Es uno de los dos códigos sustantivos que emergen en la codificación abierta. Tienden a ser comportamientos o procesos que explican cómo el problema se resuelve
3. *Memoing and Sorting Memos*. Fuerzan a determinar cómo se relaciona una categoría con otras.
4. *Models*. Representación de los códigos como entidades que se relacionan ayudan a la codificación teórica

Propone familias de códigos, el original es de 1978 y luego se hicieron agregados.

4 Tipos de datos a considerar

Algunos autores (Hoda, por ejemplo) subrayan que la *Data* [1]está formada por vuelcos de entrevistas con lo que GT reduce su campo de acción a los datos obtenidos en las entrevistas en el campo. Sin embargo esta no es la visión de la GT clásica. Glaser dice:

"Since no proof is involved, the constant comparative method in contrast to analytic induction requires only saturation of data-not consideration of all available data, nor are the data restricted to one kind of clearly defined case. The constant comparative method, unlike analytic induction, is more likely to be applied in the same study to any kind of qualitative information, including observations, interviews, documents, articles, books, and so forth. As a consequence, the constant comparisons required by both methods differ in breadth of purpose, extent of comparing, and what data and ideas are compared." [3] (pag 104).

En este párrafo hay un par de consideraciones importantes para aplicar GT:

1. CCM no requiere considerar *toda* la data.
2. Se aplica a cualquier clase de información cualitativa: observaciones, entrevistas, documentos, artículos, libros, etc.

Analizando la conducta de los sociólogos ante la data, la define como muy acotada en la consideración de documentos.

---

[1] *Data*, tiene su origen como plural del latín *datum*. Alude a un conjunto de valores cualitativos y cuantitativos. La *data* es colectada, analizada, informada, etc. Es un concepto muy común en las ciencias sociales y permite incorporar textos al análisis de las investigaciones.

“The extremely limited range of qualitative materials used by sociologists is largely due to the focus on verification. For many, if not most, researchers, qualitative data is virtually synonymous with field work and interviews, combined with whatever "background" documents may be necessary for putting the research in context. […] And so some sociologists undoubtedly have never seriously considered the library as a source of real data for their work. […] But sociologists need to be as skilled and ingenious in using documentary materials as in doing field work. These materials are as potentially valuable for generating theory as our observations and interviews. We need to be as effective as historians in the library, but with inquiry directed to our own purposes.” [3] (pags 162-163)

En definitiva GT incorpora la consideración de materiales no entrevistas.

5 Teoría

Para Laser y Strauss [3] una teoría está compuesta de dos grupos de elementos:

1. Categorías conceptuales y sus propiedades conceptuales
2. Hipótesis o relaciones generalizadas entre categorías y sus propiedades

Vale decir, se trata de dos tipos de objetos: *Categorías conceptuales* y sus *relaciones*. Ambas tienen *propiedades* y las segundas vinculan a las categorías con sus propiedades.

5.1 Categorías y propiedades

Las categorías son elementos conceptuales de la teoría y a su vez las propiedades son un aspecto conceptual de la categoría. Ambos son conceptos indicados por la data, pero no son data. Ambos tienen un nivel de abstracción diferentes del que pueden tener los datos y la variación de estos no indica la no validez de la categoría, sino que esta debe ser reemplazada por otra catego-

ría mejor: “In short, conceptual categories and properties have a life apart from the evidence that gave rise to them.” ([3], p 36). Las categorías de menor nivel aparecen en las fases iniciales de la recolección de datos, por su parte las de mayor nivel lo hacen en las fases siguientes de codificación y análisis.

En cuanto al estado del arte, es recomendable al comienzo ignorar la literatura a fin de evitar la contaminación de la teoría que emerge por conceptos previos. Glaser y Strauss creen que:

> “[…] the generation of theory should aim at achieving much *diversity* in emergent categories, synthesized at as *many levels* of conceptual and hypothetical generalization as possible. The synthesis provides readily apparent connections between data and lower and higher level conceptual abstractions of categories and properties. ([3], pag 37)

Los conceptos tienen dos características sustanciales (features):

- Los conceptos deben ser *analíticos*. Lo suficientemente generales como para definir características de entidades, no a las entidades en sí mismas
- Los conceptos deben ser *sensibilizadores* (*sensitizing*), deben sugerir una imagen significativa.

La idea de conceptos sensibilizadores se orienta a contrastar con los conceptos definitivos

> “[…] sensitizing concepts do not involve using ‘fixed and specific procedures’ to identify a set of phenomena, but instead give ‘a general sense of reference and guidance in approaching empirical instances’. So, whereas definitive concepts ‘provide prescriptions of what to see, sensitizing concepts merely suggest directions along which to look’ ([4], entrada “sensitizing concepts”, accedido 8-12-2015)

Se trata de conceptos que evocan, sugieren, abren opciones, y no cierran eventuales caminos. Producir estos conceptos requiere interactuar con gente del área. Formular estos conceptos de doble naturaleza, requiere esfuerzo de estudio de los datos y de recolección de incidentes.

5.2 Hipótesis

A través del proceso de comparación se generan categorías relaciones generalizadas entre ellas y sus propiedades [3], estas hipótesis nacen como sugerencias no verificadas y luego se consolidan. Generar hipótesis solamente requiere disponer de evidencia suficiente para establecer la sugerencia.

> "In the beginning, one's hypotheses may seem unrelated, but as categories and properties emerge, develop in abstraction, and become related, their accumulating interrelations form an integrated central theoretical framework-*the core of the emerging theory*. The core becomes a theoretical guide to the further collection and analysis of data." ([3], p 40)

5.3 Integración

La integración de la teoría tiene lugar a diferentes niveles de la teoría. Cualquiera que usa la teoría integrada debería poder "bajar" dentro de ella hasta encontrar los datos.

Type of theory		
Elements of Theory	*Substantive*	*Formal*
Category	Social loss of dying patients	Social value of people
Properties of Category	Calculating social loss on basis of learned and apparent characteristics of patient	Calculating social value of person on basis of learned and apparent characteristics
Hypotheses	*The higher the social loss of a dying patient,* 1. The better his care, 2. The more nurses develop loss rationales to explain away his death	The higher the social value of a person the less delay he experiences in receiving services from experts.

6 GT en la ingeniería de software

GT ha sido aplicado a la Ingeniería de Software, por ejemplo se utilizó para el estudio de los proceso ágiles [5], [6], [7], [8]. Es un enfoque reconocido en la investigación cualitativa en el campo de la Ingeniería de Software en general [9] y en la Ingeniería de Requerimientos en particular para sustentar la calidad de los requerimientos [10]. Ha sido utilizada en la investigación en el campo de los sistemas de información: en [11] se puede encontrar un detalle de las aplicaciones de GT en este campo de investigación a lo largo de varios años. En [12] se analizan investigaciones utilizando GT en el campo del desarrollo ágil de software, desarrollo global distribuido (GSD) y en la Ingeniería de Requerimientos. También se ha propuesta [12] un amplio campo de temas relacionados con las posibilidades de GT en el campo de la investigación en ingeniería de software. En [13] se pueden encontrar referencias a trabajos de GT relacionados con la Ingeniería de Software.

No puede omitirse que tras la capacidad de GT de desarrollar una teoría para explicar un conjunto de datos, se ha avanzado en su utilización como técnica para especificación de requerimientos [13].

7 Referencias

[1] R. Hoda, J. Noble, and S. Marshall, "Grounded Theory for Geeks," in *PLoP '11 Proceedings of the 18th Conference on Pattern Languages of Programs*, Portland, Oregon, 2011, p. Art No 24.

[2] C. A. Hernandez, "Theoretical Coding in Grounded Theory Methodology," *The Grounded Theory Review*, vol. 8, no. 3, pp. 51–66, 2009.

[3] Glaser, Barney G. and A. L. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*. New Jersey, USA: Aldine Transaction, 2008.

[4] V. Jupp, "The SAGE Dictionary of Social Research Methods," *Sage Research Methods*, 2006. [Online]. Available: https://srmo.sagepub.com/publicstart?authRejection=true. [Accessed: 12-Aug-2015].

[5] R. Hoda, J. Noble, and S. Marshall, "Developing a grounded theory to explain the practices of self-organizing Agile teams," *Empir Software Eng*, vol. 17, no. 6, pp. 609–639, Dec. 2012.

[6] R. Hoda, "Self-Organizing Agile Teams: A Grounded Theory," Ph. D. Thesis, Victoria University of Wellington, Victoria, 2011.

[7] R. Hoda, J. Noble, and S. Marshall, "Organizing Self-Organizing Teams," presented at the Software Engineering, 2010 ACM/IEEE 32nd International Conference on, Cape Town, South Africa, 2010, vol. 1, pp. 285–294.

[8] K. Power, "Stakeholder theory and agile software development," in *Proceedings of the 11th International Conference on Product Focused Software*, Limerick, Irlanda, 2010.

[9] D. I. K. Sjøberg, T. Dybå, B. C. D. Anda, and J. E. Hannay, "Building Theories in Software Engineering," in *Guide to Advanced Empirical Software Engineering*, Springer-Verlag London Limited, 2008, pp. 312–336.

[10] Power, Norah, "A Grounded Theory of Software Requirements Quality," in *Proceeding RE95*, York, England, 1995, p. 4.

[11] R. Matavire and I. Brown, "Investigating the Use of 'Grounded Theory' in Information Systems Research," in *Proceedings of the 2008 annual research conference of the South African Institute of Computer Scientists and Information Technologists on IT research in developing countries: riding the wave of technology*, Wilderness, South Africa, 2008, pp. 139–147.

[12] O. Badreddin, "Thematic Review and Analysis of Grounded Theory Application in Software Engineering," *Advances in Software Engineering*, vol. 2013, no. 468021, p. 9, 2013.

[13] D. M. Berry, M. W. Godfrey, R. Holt, C. J. Kapser, and I. Ramos, "Requirements Specifications and Recovered Architectures as Grounded Theories," *The Grounded Theory Review*, vol. 12, no. 1, pp. 56–65, Jun. 2013.