INTERNATIONAL DESIGN CONFERENCE - DESIGN 2016
Dubrovnik - Croatia, May 16 - 19, 2016.

DESIGNING IN A UNIVERSITY AND START-UP CONTEXT: A STUDY OF THE DEPENDENCY BETWEEN DESIGN REQUIREMENTS

R. A. Duran-Novoa and E. C. Y. Koh

Keywords: requirements, engineering change, entrepreneurship, change prediction

1. Introduction

The accurate management of requirements plays a critical role in product development, since missing any of them could lead to avoidable rework or even malfunctions [Chen and Zeng 2006], [Jarratt et al. 2011], [Hamraz et al. 2013]. This is especially dangerous in start-up organizations: untested market conditions, regulatory obstacles, and changing technology are examples of characteristics that make future system performance and success very challenging over the long term [Lessio et al. 2015].
In order to focus the limited resources in the correct requirements, it is necessary to understand how they affect the components and how they affect each other. The requirement-component dependencies have been studied widely [Koh et al. 2012, 2013, 2015], [Tang et al. 2015], [Tang and Yin 2016]; however, the research about requirement-requirement dependencies is almost inexistent, probably because they are more difficult to determine and quantify since (1) requirements are usually written in natural language, which leads to ambiguous or distorted understanding [Chen and Zeng 2006], and (2) requirements are volatile, having the tendency of changing over time [Nurmuliani et al. 2004].
Aiming to improve the understanding of dependencies between requirements, we contribute with two indices that indirectly quantifies them (Section 2), exploring the way that requirements affect each other and how to deal with this in a dynamic context, topic not treated (to our knowledge) in the literature. These indices were applied in a study case within a start-up context (Section 3), to later analyse the results and discuss its usefulness (Section 4). Conclusions and future work directions are presented at the end of the paper (Section 5).

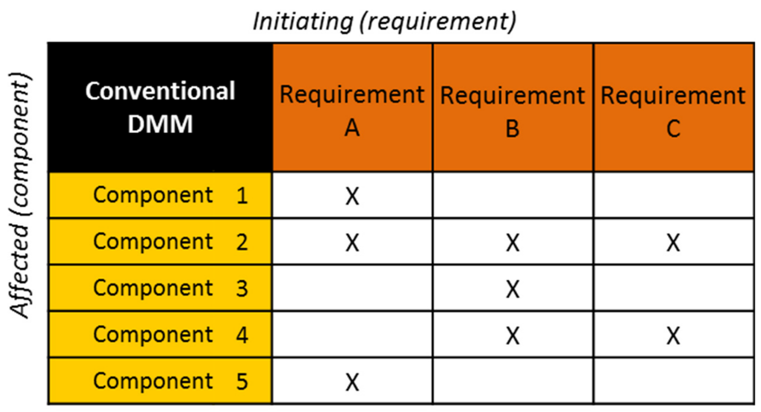
2. Understanding dependencies between design requirements

To understand how requirements affect each other, we need an operational definition. Jacobson in "The Unified Software Development Process" defines requirement as "a condition or capability to which a system must conform; either derived directly from user needs, or stated in a contract, standard, specification, or other formally imposed document" [Jacobson et al. 1999], while Grady in "System Requirements Analysis" declares that "requirements are the necessary attributes defined for a system before design development. The customer's need is the ultimate system requirement from which all other requirements and designs flow" [Grady 1993]. We consider that to understand how requirements affect each other it is useful to distinguish two sub-categories: qualities and constrains. Qualities are derived from the user needs and are conceptual in nature, while constrains are specific declarations that the design must comply due to its context (e.g. laws, norms, budget). Based on the previous, we define requirement as "any attribute or condition that a system must have (quality) or fulfil (constrain)". For

example, a car requirement could be ‘display the logo of the company’, which should be considered a constrain, since the logo does not affect the transport capacity to the car -its main purpose- but it has to be fulfilled.
This distinction has the potential of being useful when preparing for changes. Between constrains, due to its specific nature, it is reasonable to suppose how they will affect each other from the moment that they are defined. For example, a car manufacturer could anticipate most of the effects that the ‘Euro V emission standard’ will have on the ‘exhaust pipe development budget’. However, when a quality is involved the situation is different, since their listing is ambiguous and they tend to be interrelated in unexpected ways between each other. For example, it is possible to anticipate how the ‘elegance’ of a car will affect its ‘safety’ or the ‘exhaust pipe development budget’?
A related work that discuss dependencies between requirements was done by [Morkos et al. 2012], but their study used the information provided by a well-documented process to determine if change propagation can be predicted using higher order DSM, while our focus is how to deal with the dynamic relations of requirements during design.

2.1 Previous to modelling: requirement-component dependencies

In order to construct our model, it was necessary to establish the requirement-component dependencies, which we did adapting a Domain Mapping Matrix (DMM) [Danilovic and Browning 2004]. A DMM presents in a rectangular matrix the dependencies between elements of different domains, in our case between requirements and components (as presented in Figure 1). This format allows users to visualise simultaneously multiple relations and thus extract patterns or tendencies hidden inside the complexity. For example, a quick look at Figure 1 reveals that requirement C is the one that affect fewer components, and that component 2 is the most affected one.


Initiating (requirement)

Affected (component)

Conventional DMM	Requirement A	Requirement B	Requirement C
Component 1	X		
Component 2	X	X	X
Component 3		X	
Component 4		X	X
Component 5	X		

Figure 1. Example of a classic DMM matrix

2.2 Modelling dependencies between design requirements

Ideally, requirements should be as independent as possible to avoid change propagation; however, some level of influence is unavoidable since they share the same design purpose. For example, two phone independent requirements could be ‘being attractive’ and ‘high reception quality’; but if an aesthetic detail interferes with the reception, the phone ability to communicate –its final purpose- will determine the level of compromise that can be made.
To quantify the potential influence between requirements we propose two indices: the “coincidences with other requirements” and the “coincidences ÷ dependencies” ratio. Both are derived from the requirement-component dependencies, since its magnitude determines the potential effect that the modification of a requirement can have on the design.
The “coincidences with other requirements” is an absolute index that shows the total number of coincidences between a requirement and the other requirements. Figure 2 -constructed from Figure 1 data- shows these coincidences in a square matrix, where the intersection of two requirements indicates the number of components that they affect simultaneously. For example, requirement A has 1

coincidence with B and 1 with C, giving a requirement coincidence index of 2. A bigger number of coincidences increase the chances of dependency.
The "coincidences ÷ dependencies" ratio is a relative measurement of the influence that requirements can potentially have on each other. It consists on the division of the "requirement coincidences" by the quantity of requirement-component dependencies, which are included in the grey cells of the diagonal. For example, in Figure 2 requirement A has an index of 0.7, while Requirement C has an index of 1.5, suggesting that requirement C links are stronger.

	Requirements				
	A	B	C		
A	3	1	1	Req-component dependencies	3
				Coincidences with other reqs.	2
				Coincidences ÷ Dependencies	0.7
B	1	3	2	Req-component dependencies	3
				Coincidences with other reqs.	3
				Coincidences ÷ Dependencies	1.0
C	1	2	2	Req-component dependencies	2
				Coincidences with other reqs.	3
				Coincidences ÷ Dependencies	1.5

Figure 2. Example of requirement indices

The proposed indices are statistically-based expressions that display coincidences and correlation between requirements, and consequently they cannot be used to establish if two requirements are correlated: the coincidences between requirement-component dependencies can be caused by a third requirement, the combination of other variables, or just chance. However, since these coincidences occur in a closed system, we should be able to observe tendencies that help us decide where to analyse further, diminishing the trials and errors and thus changes. For example, requirement C seems to be more relevant than A, because it shows higher indices despite having lower requirement-component dependencies. Now regarding the independence of requirements, the indices can give us a stronger result: if it has a lower number of coincidences (like requirement A in Figure 2), we can conclude that the requirement is relatively more independent.

3. A case study on an electrical motorcycle modification project

Our particular case study is a university project conducted by a multi-disciplinary team under the guidance and demand of industry experts, as a part of the Design Centric Program of the National University of Singapore [Loh 2015]. We consider it a valid representation of a start-up because the university requires from students to display the professional abilities demanded by a changing industry (e.g. technical knowledge, communication skills, problem solving abilities, teamwork, critical thinking, calculated risks, performance evaluation), while the project itself has to deal with the characteristics of any start-up: it begins from scratch, any change have a big impact, experience is limited, there is great flexibility, the investors are critical, and the negotiation power is minimal [Bommer and Jalajas 2004], [Klepper 2007], [Song et al. 2010], [Lau et al. 2010], [Marion et al. 2012], [Christensen 2013], [Song et al. 2014]. Specifically, the project consisted in the modification of a standard combustion motorcycle towards electrical drive. The modified motorcycle ("E-Bike") participated in the "2014 World Advance Vehicle Expedition, WAVE", obtaining the 3rd place on its category.

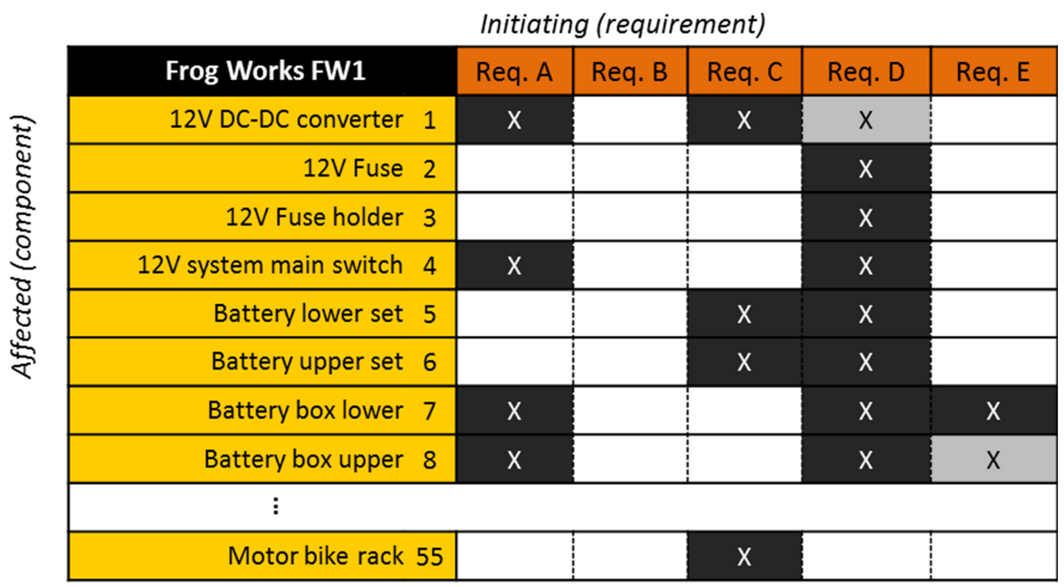
3.1 Data collection

In order to determine the changes to be made, the team listed and grouped the requirements and involved components, establishing the requirement-component dependencies. Later, every requirement was mapped into viable design options, which lead to the identification of the affected components using a DMM matrix (Figure 3).

The design requirements identified were:

- Requirement A: Provide a 'professional' engineering look with the electrical drive installed
- Requirement B: Add a Data Acquisition System
- Requirement C: Increase range and reliability
- Requirement D: Increase driving performance
- Requirement E: Replace damaged parts when required

The DMM matrix was frequently updated, registering the evolution of dependencies through time. In parallel, the coincidences between requirement-component dependencies were summarized in a square requirement matrix following Figure 2 logic (see Figures 4 and 5 in section 3.2).



Initiating (requirement)

Affected (component)

Frog Works FW1	Req. A	Req. B	Req. C	Req. D	Req. E
12V DC-DC converter 1	X		X	X	
12V Fuse 2				X	
12V Fuse holder 3				X	
12V system main switch 4	X			X	
Battery lower set 5			X	X	
Battery upper set 6			X	X	
Battery box lower 7	X			X	X
Battery box upper 8	X			X	X
⋮					
Motor bike rack 55			X		

Figure 3. Extract of DMM dependencies between requirements and components, at January 26, 2014 (final report). Light grey cells indicate pending tasks, while dark ones are finished ones

3.2 Results

The following images presents the "requirement dependency matrix" at the beginning and at the end of the project (Figures 4 and 5 respectively), including their indices normalized to facilitate analysis. It should be considered that requirement E had no dependencies associated until September 20, 2014, reason why all its original values were zero or undetermined.

In general, the indices in the original matrix were similar between requirements. For example, the number of coincidences did not surpass 4, and the coincidences ÷ dependencies ratio varied between 0.2 and 0.4 (17% and 33% respectively). This similarity disappears in final matrix: the number of coincidences varied from 4 to 36, while the coincidences ÷ dependencies ratio did it between 0.4 and 1.8 (7% and 35% respectively). This range expansion suggests that requirements are not only subjected to volatility, but also polarization.

		Requirements, 2013-09-04							
		A	B	C	D	E	Indices		Normalized
Provide a 'professional' engineering look	A	19	1	2	1	0	Req-component dependencies	19	0.51
							Coincidences with other reqs.	4	0.40
							Coincidences ÷ Dependencies	0.2	0.17
Add a Data Acquisition System	B	1	6	0	1	0	Req-component dependencies	6	0.16
							Coincidences with other reqs.	2	0.20
							Coincidences ÷ Dependencies	0.3	0.27
Increase range and reliability	C	2	0	7	0	0	Req-component dependencies	7	0.19
							Coincidences with other reqs.	2	0.20
							Coincidences ÷ Dependencies	0.3	0.23
Increase driving performance	D	1	1	0	5	0	Req-component dependencies	5	0.14
							Coincidences with other reqs.	2	0.20
							Coincidences ÷ Dependencies	0.4	0.33
Replace damaged parts when required	E	0	0	0	0	0	Req-component dependencies	0	0.00
							Coincidences with other reqs.	0	0.00
							Coincidences ÷ Dependencies	--	0.00

Figure 4. Initial "requirement dependency matrix" of the project (September 4, 2013)

		Requirements, 2015-01-26							
		A	B	C	D	E	Indices		Normalized
Provide a 'professional' engineering look	A	23	1	5	18	11	Req-component dependencies	23	0.19
							Coincidences with other reqs.	35	0.32
							Coincidences ÷ Dependencies	1.5	0.31
Add a Data Acquisition System	B	1	11	1	1	1	Req-component dependencies	11	0.09
							Coincidences with other reqs.	4	0.04
							Coincidences ÷ Dependencies	0.4	0.07
Increase range and reliability	C	5	1	21	8	0	Req-component dependencies	21	0.17
							Coincidences with other reqs.	14	0.13
							Coincidences ÷ Dependencies	0.7	0.13
Increase driving performance	D	18	1	8	54	9	Req-component dependencies	54	0.45
							Coincidences with other reqs.	36	0.33
							Coincidences ÷ Dependencies	0.7	0.13
Replace damaged parts when required	E	11	1	0	9	12	Req-component dependencies	12	0.10
							Coincidences with other reqs.	21	0.19
							Coincidences ÷ Dependencies	1.8	0.35

Figure 5. Final "requirement dependency matrix" of the project (January 26, 2013)

Table 1 shows the indices values including the percent variation. This allows visualizing, for example, that in absolute terms their magnitudes augmented considerably, but in relative terms they tended to diminish. This can be explained mostly by the late inclusion of requirement E and the redistribution caused by the growth of requirement D (its number of coincidences augmented a 1700% in absolute terms, but "only" 64% regarding the total).

Looking at the final indices several observations can be made: requirement A is probably the most influential, since both of its indices are high; requirement B is by far the most independent, since all their indices are the lowest; requirement C seems to have a moderate influence, probably due to its more technical nature; requirement D has a big influence due to its large number of requirement-component dependencies; and requirement E is strongly connected to the other requirements despite having a low number of requirement-component dependencies, which is explained by its strictly dependant nature (something has to fail to be repaired).

Table 1. Requirements and their indices variation

Requirements		Indices	Initial		Final		Percent variation	
			Value	Norm.	Value	Norm.	Absolute[1]	Relative[2]
A	Provide a 'professional' engineering look	Req-component dependencies	19	0.51	23	0.19	21%	-63%
		Coincidences with other reqs.	4	0.40	35	0.32	775%	-20%
		Coincidences ÷ Dependencies	0.2	0.17	1.52	0.31	623%	79%
B	Add a Data Acquisition System	Req-component dependencies	6	0.16	11	0.09	83%	-44%
		Coincidences with other reqs.	2	0.20	4	0.04	100%	-82%
		Coincidences ÷ Dependencies	0.3	0.27	0.36	0.07	9%	-73%
C	Increase range and reliability	Req-component dependencies	7	0.19	21	0.17	200%	- 8%
		Coincidences with other reqs.	2	0.20	14	0.13	600%	-36%
		Coincidences ÷ Dependencies	0.3	0.23	0.67	0.13	133%	-42%
D	Increase driving performance	Req-component dependencies	5	0.14	54	0.45	980%	230%
		Coincidences with other reqs.	2	0.20	36	0.33	1700%	64%
		Coincidences ÷ Dependencies	0.4	0.33	0.67	0.13	67%	-59%
E	Replace damaged parts when required	Req-component dependencies	0	0.00	12	0.10	--	--
		Coincidences with other reqs.	0	0.00	21	0.19	--	--
		Coincidences ÷ Dependencies	--	0.00	1.75	0.35	--	--

[1] Percent variation of the requirement index regarding itself

[2] Percent variation of the requirement index regarding the total

Table 2 shows the ranking of each index at the beginning and at the end of the project. This general view helps to determine priorities, but it must be considered that positions are not proportional to the correspondent index, reason why the normalized value is also included.

Table 2. Ranking of requirements indices

Requirement	Indices	Initial rank		Final rank	
A	Req-component dependencies	1	(0.51)	2	(0.19)
	Coincidences with other reqs.	1	(0.40)	2	(0.32)
	Coincidences ÷ Dependencies	4	(0.17)	2	(0.31)
B	Req-component dependencies	3	(0.16)	5	(0.09)
	Coincidences with other reqs.	2-3-4	(0.20)	5	(0.04)
	Coincidences ÷ Dependencies	2	(0.27)	5	(0.07)
C	Req-component dependencies	2	(0.19)	3	(0.17)
	Coincidences with other reqs.	2-3-4	(0.20)	4	(0.13)
	Coincidences ÷ Dependencies	3	(0.23)	3-4	(0.13)
D	Req-component dependencies	4	(0.14)	1	(0.45)
	Coincidences with other reqs.	2-3-4	(0.20)	1	(0.33)
	Coincidences ÷ Dependencies	1	(0.33)	3-4	(0.13)
E	Req-component dependencies	5	(0.00)	4	(0.10)
	Coincidences with other reqs.	5	(0.00)	3	(0.19)
	Coincidences ÷ Dependencies	5	(0.00)	1	(0.35)

4. Discussion

We observed volatility and polarization in the dependencies: initially, the indices values were similar, but then they changed at different rates which resulted in distant results. For example, requirement B was initially relevant, but in the final report its independency was visible, while requirement D variation was almost the opposite. This confirms that, at least in the studied context, initial predictions of requirement-requirement dependencies are unlikely to be reliable.

The previous shows the importance of visualizing how requirements dependencies evolve during a project, in order to take action and develop a correct sequence of activities that minimize the change propagation. For example, if the E-Bike project started to work in the next model, a reasonable plan could start by solving requirement B in an independent module, and focus most of the efforts on requirements A and D simultaneously; since A and D changes will probably affect requirements C and E, they (C and E) should be maintained in observation until the polarization and volatility reduce its magnitude. This proposal can be made even without knowing the nature of the requirements, and it is coherent with reality: requirement A (Provide a 'professional' engineering look with the electrical drive installed) is transversal and likely to influence many components; requirement B (Add a Data Acquisition System) can be develop in parallel by stablishing its input and output; requirement C (Increase range and reliability) is linked to the other requirements, but within clear limits; requirement D (Increase driving performance) seemed to be moderately related, but it "hides" all the legal needs that makes it so connected; and finally requirement E (Replace damaged parts when required) is a consequence of the other requirements that could be delayed as much as possible.

In brief, the proposed indices point towards the most influential requirements, allowing stablishing hierarchies and determining where a deeper analysis could improve an ongoing project.

5. Conclusions and future work

It is unlikely to anticipate how requirements will affect each other, at least in high-variability environments like start-ups, due to their volatility and polarization. Since they determine changes and rework, to quantify their relative importance can help managing unexpected scenarios.

The proposed indices were a valuable tool to understand the dependencies between requirements (requirement-requirement dependencies), showing consistency between their values and what was observed in practice (the E-Bike project). This suggest that they can deliver a low-prejudge analysis that help to visualize tendencies during the development of a project, in order to focus the resources in the most influential requirements, even without completely understanding them.

In future research, two topics will be developed based on this paper. First, to determine if the type of requirement –qualities or constrains- has a practical effect on volatility, in order to stablish a correct sequence of analysis. Second, to define the conditions needed to obtain useful results in complex systems, where a "manual" analysis is not plausible due to the great amount of data.

Acknowledgement

The authors would like to thank the members of the Design-Centric Programme and the University Scholars Programme at the National University of Singapore for their valuable support. This research is funded by the EDIC Seed Fund (R-261-503-003-133), National University of Singapore.

References

Bommer, M., Jalajas, D. S., "Innovation Sources of Large and Small Technology-Based Firms", IEEE Transactions on Engineering Management, Vol.51, No.1, 2004, pp. 13–18.

Chen, Z. Y., Zeng, Y., "Classification of Product Requirements Based on Product Environment", Concurrent Engineering, Vol.14, No.3, 2006, pp. 219–230.

Christensen, C., "The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail", Harvard Business Review Press, 2013.

Danilovic, M., Browning, T., "A Formal Approach for Domain Mapping Matrices (DMM) to Complement Design Structure Matrices (DSM)", The Sixth Design Structure Matrix (DSM) International Workshop, 2004, pp. 1–23.

Grady, J. O., "System Requirements Analysis", Mcgraw-Hill, Now York, USA, 1993.

Hamraz, B., Caldwell, N. H. M., Clarkson, P. J., "A Holistic Categorization Framework for Literature on Engineering Change Management", Systems Engineering, Vol.16, No.4, 2013, pp. 473–505.

Jacobson, I., Booch, G., Rumbaugh, J., "The Unified Software Development Process", Vol.1, Addison-wesley, 1999.

Jarratt, T., Eckert, C. M., Caldwell, N. H. M., Clarkson, P. J., "Engineering Change: An Overview and Perspective on the Literature", Research in Engineering Design, Vol.22, No.2, 2011, pp. 103–124.

Klepper, S., "Disagreements, Spinoffs, and the Evolution of Detroit as the Capital of the US Automobile Industry", Management Science, Vol.53, No.4, 2007, pp. 616–631.

Koh, E. C. Y., Caldwell, N. H. M., Clarkson, P. J., "A Technique to Assess the Changeability of Complex Engineering Systems", Journal of Engineering Design, Vol.24, No.7, 2013, pp. 477–498.

Koh, E. C. Y., Caldwell, N. H. M., Clarkson, P. J., "A Method to Assess the Effects of Engineering Change Propagation", Research in Engineering Design, Vol.23, No.4, 2012, pp. 329–351.

Koh, E., Förg, A., Kreimeyer, M., Lienkamp, M., "Using Engineering Change Forecast to Prioritise Component Modularisation", Research in Engineering Design, Vol.26, No.4, 2015, pp. 337–353.

Lau, A. K. W., Tang, E., Yam, R., "Effects of Supplier and Customer Integration on Product Innovation and Performance: Empirical Evidence in Hong Kong Manufacturers", Journal of Product Innovation Management, Vol.27, No.5, 2010, pp. 761–777.

Lessio, M. P., Cardin, M. A., Astaman, A., Djie, V., "A Process to Analyze Strategic Design and Management Decisions Under Uncertainty in Complex Entrepreneurial Systems", Systems Engineering, Vol.18, No.6, 2015, pp. 604–624.

Loh, A. P., "The Design Centric Programme at the National University of Singapore", Proceedings of the I-PHEX 2015 'Innovative Practices in Higher Education Expo 2015', Universiti Teknologi Malaysia, Malaysia, 2015.

Marion, T., Dunlap, D., Friar, J., "Instilling the Entrepreneurial Spirit in Your R&D Team: What Large Firms Can Learn from Successful Start-Ups", IEEE Transactions on Engineering Management, Vol.59, No.2, 2012, pp. 323–337.

Morkos, B., Shankar, P., Summers, J. D., "Predicting Requirement Change Propagation, Using Higher Order Design Structure Matrices: An Industry Case Study", Journal of Engineering Design, Vol.23, No.12, 2012, pp. 905–926.

Nurmuliani, N., Zowghi, D., Fowell, S., "Analysis of Requirements Volatility during Software Development Life Cycle", Proceedings of the 2004 Australian Software Engineering Conference, 2004, pp. 28–37.

Song, C.-Y., Luo, J., Holtta-Otto, K., Otto, K., "Product Innovation Differences between New Ventures and Incumbent Firms", Academy of Management Proceedings 2014, Vol.1, 2014.

Song, L. Z., Song, M., Parry, M. E., "Perspective: Economic Conditions, Entrepreneurship, First-Product Development, and New Venture Success", Journal of Product Innovation Management, Vol.27, No.1, 2010, pp. 130–135.

Tang, D. B., Yin, L. L., "Using an Engineering Change Propagation Method to Support Aircraft Assembly Tooling Design", Proceedings of the 6th International Asia Conference on Industrial Engineering and Management Innovation, 2016, pp. 939–951.

Tang, D. B., Yin, L. L., Wang, Q., Ullah, I., Zhu, H. H., Leng, S., "Workload-Based Change Propagation Analysis in Engineering Design", Concurrent Engineering, 2015.

Dr.-Ing. Roberto Alejandro Duran-Novoa, Research fellow
National University of Singapore, Engineering Design and Innovation Centre
Block E2A #02-03, 5 Engineering Drive 2, 117579 Singapore , Singapore
Email: engradn@nus.edu.sg