A Survey on the Interplay between Software Engineering and Systems Engineering during SoS Architecting - Quotes of the codes, with most occurrences, induced from the responses to the open-ended questions.

1 Question 3.1.

Table 1: Prominent codes that emerged from the *Qualitaitive Content Analysis* process on the responses to the second part of question 3.1.

Code	**Quote**	**# ocurrences in the question.**
TEAMS COORDINATION CHALLENGES	(R10) *"Communications between systems and software engineers"*, (R31) *" It is challenging to keep engineering teams' perspectives in mind when coordinating across teams."*, (R1) *The integration of the teams and whole goals communication*, (R24) *Poor coordination, understanding, of dominant requirements—whether originating with software or with system concerns.*	4
INCOMPLETE SYSTEM REQUIREMENTS	(R2) *"Poorly documented or incomplete elicitation results and stakeholder analysis."*, (R60) *"'Vagueness' in system level requirement (when train is at stop) leaves lot of interpretation on how software determines actual state from odometry captors. "*, (R41) *Quality of Customer Requirements not sufficient, Several NFRs unclear/unspecified, Functional Safety Requirements added very late*, (R50) *Scarce description of requirements on system level.*	4
ASSUMPTIONS ABOUT THE OTHER DISCIPLINE	(R20) *"The focus was on the subsystem level with the assumption that the system level issues could just be resolved as they came up..."*, (R25) *" Assumptions were made by systems engineers regarding where to allocate software components, which led to timing/sizing and reliability concerns."*, (R31) *...Many software engineers assumed that higher-level requirements existed and wanted to be guided by them. Systems engineers often assumed that software had not/could not have been properly developed without higher-level requirements...* , (R30) *System Level requirements posed significant limits on the software and hardware infrastructure required, which did not allow the facilitation of infrastructure best suited for the project...*	4
INTERDISCIPLINARY DIFFERENCES	(R25) *"System, software, and programmatic (cost/schedule) goals of the architecture were sometimes in conflict"*, (R30) *"System Level requirements posed significant limits on the software and hardware infrastructure required"*, (R54) *Decomposition of the system was different from system (more mechanical) and software.*, (R8) *Collaborative Architecting activities involving several nations and several companies towards a single solution being a System of Systems.*	4

Continued on next page

Table 1 – continued from previous page

Code	Quote	# ocurrences in the question.
LACK OF SYSTEM-LEVEL PERSPECTIVE	(R17) *"... the system architecture was largely ignored by the electrical and software teams when they architected the software and electrical systems, so there was a change in coupling and interdependence from the original system architecture..."*, (R20) *"...the problems at the system level were very difficult to solve and caused major delays and program replans. Eventually, the program was given to a different contractor who understood the need for systems engineering..."*, (R16) *Gaining a shared, and agreed, understanding of a desired level of abstraction an granularity for respective architectures.i.e. Systems imposing levels of granularity that constrict (rather than constrain) software architecture.*	3
LACK-OF-DOMAIN-KNOWLEDGE	(R38) *"Lack of domain knowledge and business understanding in SW-Development leads to wrong interpretation of requirements..."*, (R11) *Common understanding among involved disciplines, even from the same domain*, (R9) *... lack of experience of software engineers related to avionics systems*	3

2 Question 3.6.

Table 2: Prominent codes that emerged from the *Qualitaitive Content Analysis* process on the responses to question 3.6

Code	Quote	# ocurrences in the question.
ARCHITECTURE-PATCHING	(R2) *"Too late, much effort needed to make bandaid patches."*, (R12) *Too late, much effort needed to make bandaid patches...*, (R14) *'patches and problems not completely solved but considered as solved.*, (R17) *Flaws in the architecture were discovered deep into implementation, and so there was little tolerance for rearchitecting due to schedule and budget impact.* (R30) *"There was no clear separation between software-level and system-level architecture, thus the evaluation process sometimes resulted in changes to the system-architecture and sometimes to functional requirements on the software level."*	5
INSUFFICIENT-EVALUATION	(R50) *"Overall insufficient evaluation."*, (R53) *"More lessons could have been learned by better evaluation."*, (R19) *"Evaluation seemed to be informal."*, (R24)"Coverage was spotty."	5
EVALUATION-COMPLEXITY	(R24) *"Overly detailed and complex: more like plumbing and wiring diagrams than architectures!"*, (R10) *"Integration, Understanding of processes"*, (R21) *"The system-level architecture evaluation required some training and going through the actual proces"*	3
EVALUATION-CRITERIA	(R8) *"very difficult to get a set of weighted criteria agreed by the Stakeholder"*, *"changes derived from the evaluation, lack of traceabilty, architecture patches"*, (R1) *"Evaluate criteria, is the big challenge to define consider both aspects [system/software] on the early phases"*	2

3 Question 4.2.

Table 3: Prominent codes that emerged from the *Qualitaitive Content Analysis* process on the responses to question 4.2

Code	**Quote**	**# ocurrences in the question.**
M&S APPROACHES LIMITATIONS	(R27) *Impossible to model all possible emergent behaviour, particularly when dealing with large scale systems with multiple integrations, sub-systems, sensors and data models.* (R19) *Lab testing was much less likely to show any emergent behavior due to the lack of realistic testing environments.* (R60) *System uses wireless communication that has important error rate, link from radio performance to visible system impact (train emergency braking) very difficult to characterize.*	4
SUBSYSTEMS-RELATED-EMERGENCE-CAUSES	(R23) *COTS selected for subsystems too ridged and did not support emerging needs well. Limited ability to support emerging needs.*, (R25) *Emergent behaviors in this case were primarily caused by faulty components or low-level requirements, not architecture.*, (R21) *use of shared resources by competing systems*, (R3) *Timing of hardware components. Undocumented backdoors.*	4
TIME-AND-RESOURCES-PROBLEMS	(R12) Added cost - for systems to patch sw arch holes. (R9) Ongoing rework. Defect growth extending post repeated projection over life cycle. Exceeding projected demand on software processor resources, putting safety at risk. (R38) Not enough time and resources sent for "architecture governance". (R24) There was a recognition (on both system and software teams) of the need to manage emergences, but aside from some specific cases, neither team had adequate means to do so.	4
UNSATISFIED REQUIREMENTS	(R18) *Available software, driven by the system level decisions, did not provide expected functionality in the aggregate*, (R20) *If undesired emergent behavior includes not meeting key system requirements, this led to the program being taken away from this contractor and given to another one.*	2

4 Question 4.5.

Table 4: Prominent codes that emerged from the *Qualitaitive Content Analysis* process on the responses to the third part of question 4.5.

Code	**Quote**	**# ocurrences in the question.**
CONFIGURATION MANAGEMENT	(R1) *"Configuration Management is a challenge to this subjects"*, (R8) *Stability of the interface specifications with regards to evolution of the systems*, (R34) *"Specifications are not enough, you need lifecycle policies. How long will a specification be supported? How are consumers notified of (upcoming) changes? What is the backward compatibility time window?"*, (R24) *For constituent systems in current development, the interface specs always lagged reality....*, (R31) *...documentation and version control of interface documentation was an issue.*, (R41) *... Some changes were actually required due to improper interface specification, so the situation actually improved. However, close synchronization was required between the affected teams.*	6
INCOMPLETE INTERFACE SPECIFICATIONS	(R20) *"Evolution of the subsystems did not affect changes to the interface specifications as the specifications were not adequate and the system impacts of subsystem changes were not appreciated."*, (R37) *"Missing clarity on intended usage of APIs - more docs and examples improved"*, (R41) *"Some changes were actually required due to improper interface specification, so the situation actually improved. However, close synchronization was required between the affected teams."*, (R51) *"Specification not complete to consider all cases."*	4
HARDWARE-SOFTWARE CHANGES BALANCE	(R9) *"Dependence on software to address hardware interface issues."*, (R12) *" Cannot solve all in software, need a balanced solution."*,	2
CHANGE COSTS	(R33) *" time waste in the review, validation and changes, into the neighbor sub-systems, to adapt them to the new interface"*, (R38) *"Changes often result claims requiring addtional efforts (at least financial), so more stakeholders get involved making the change-process slower or blocking it"*	2