VBB LeSS Adoption Case Study

Background and History 2
Organizational Design and Descaling 2
Organizational Culture 4
Adoption of LeSS 5
  Recognizing Pre-conditions Supportive of LeSS Adoption 5
  Learning Lessons from Adopting Basic Scrum 6
  Defining (and Expanding) Product Definition 10
  Realigning Teams (from Sub-Systems to One Product) 11
  Identifying Product Owner and SMEs 14
  Expanding LeSS by Adding More Teams 17
  Improving Overall Interaction with Business 19
  Minimizing Side-work and Supporting Single Product Backlog 20
  Providing a Non-hierarchy Based Escalation Mechanism 21
  Improving Engineering Practices 21
  Forming Communities of Practice 24
  “Narrowing the Gap between Science and Business” (Daniel Pink) 25
  Introducing Agile Budgeting 28
  Flattening Overall Organization Design 29
Conclusion 31
Background and History

VBB - Is a large investment company, with thousands of locations and hundreds of thousands of employees around the globe. During almost 150 years of its existence, the company has gone through many phases of organizational development and restructuring. Today, it is rightfully considered, as one of the biggest and most successful financial institutions in the world.

This case study is about 2015-2016 Large Scale Scrum (LeSS) adoption experience, shared by the two agile coaches: Stuart Paterson and Gene Gendel. Cumulatively, the coaches had experience in: organizational design, DevOps, agile training and coaching, at team and enterprise level. More specifically, Gene brought to the table his experience of working with individuals, teams and senior leadership, with focus on organizational design, system dynamics, norms, policies, practices and procedures, that he gained as a coach-consultant, serving multiple organizations over a decade. Stuart - came with rich experience of software engineering, development practices, test automation, TDD, CI/CD, as well as deep understanding of VBB's internal dynamics that he has acquired over years. From the onset of this engagement, both Gene and Stuart felt that their skills and experience were complementary and would help them achieve a common goal.

Organizational Design and Descaling

First, let's take a look at the very high level organizational design of the company, as it existed before LeSS adoption.

The following acronyms/abbreviations will be used throughout the case study, when referring to various organizational structures:

- VBB - Very Big Bank; the whole company
- CTO domain - large organizational area, within VBB, controlled by one CTO
- CTO-Biz Partner domain - large organizational area, within VBB, controlled by a senior leader - CTO’s counterpart
- TGIF - Technology Group In-Flight - organizational area within CTO Domain, where LeSS adoption was experimented
• BGIF - Business Group In-Flight - organizational area within CTO-Biz Partner Domain, also involved in LeSS adoption. There was also an extended business community (bigger than BGIF) that was not directly involved in LeSS adoption.
• LeSS Construct - people from both, TGIF and BGIF that were involved in LeSS adoption (Note: LeSS Construct = LeSS “organizational sushi roll” - represents multiple layers of the organizational structure, not just IT).

Historically VBB - was a classic multi-layered organizational structure, with multiple reporting layers.
On technology side, there were at least 3-4 reporting layers of management, between higher decision makers (CIO/CTO) and real doers (hands-on developers).
On business side, VBB looked something similar: with multiple “translators” and “delegates”, such as BAs, PMO and various engagement managers.

The coaches belonged to the same organizational structure, spearheaded by the same CTO, but fortunately, had full access to all layers of CTO domain, as well as laterally: into CTO-Biz Partner domain.
In order to succeed with LeSS adoption, the coaches had to influence both: CTO and CTO Biz Partner domains. For that, the coaches tried the following LeSS organizational design experiment: “Try… Keep the organization as flat as possible 241 • Try… Make the organization slightly flatter than it can handle. 242”.

Instead of trying to flip the whole organization at once (TGIF + BGIF), the coaches decided to do it incrementally, by first identifying LeSS “Sushi Roll” - a construct that would consist of people, on technology and business side (TGIF and BGIF, respectively) involved in LeSS adoption, and then gradually coach the organization towards flattening. For example, on technology side, it would mean going from vertically-stretched secluded towers, made of component teams and applications groups to flatter CTO-governed domains, made of feature teams - teams that could work on an entire customer-centric feature, cutting through multiple components and application layers.

Graphically, LeSS Construct, looked like the following, inside VBB: it brought together and ring-fenced (protected), flattened organizational domains from technology and business.
One of the most challenging aspects of organizational culture that the coaches had to face was the amount of emphasis that was put on individual performance. This was not only problematic for successful LeSS (or even basic Scrum) adoption. There was a much deeper systemic problem that had to be addressed, organization-wide. Historically, VBB, had the culture that encouraged super-heroics and internal competition. People were primarily driven by extrinsic motivation and a desire to outperform their colleagues, competing for bonuses, promotions and other perks. Individual performance appraisals and end-of-year reviews defined individuals' behaviors and often led to system gaming, especially at year-end.

As an example, the below causal loop diagram describes some underlying system dynamics that involved individual performance evaluations and its impact on basic Scrum dynamics.
To alleviate the above described problem within LeSS Construct, the coaches decided to leverage the following LeSS experiment: "Avoid… Incentives linked to performance 268", and "Try… De-emphasize incentives 270". More about this effort below.

Adoption of LeSS

Recognizing Pre-conditions Supportive of LeSS Adoption

The following, were some internal preconditions within TGIF that were already supportive of LeSS guides and experiments before the coaches got engaged:

- PMO, as an organizational unit, was dissolved and some remaining PMO-like functions were reabsorbed by teams and business people (LeSS experiment: "Avoid… Project Management Office 249, Avoid… So-called Agile PMO 249")
- Already, there was clear understanding, at all organizational levels, that having Scrum Masters doing performance appraisals on team members, would be a sign of serious dysfunction (LeSS experiment: "Avoid… Scrum Masters do performance appraisals 275")
● Even with initial adoption of basic Scrum (before LeSS), teams strived to expand their respective (individual) Definition of Done, with each sprint (LeSS experiment: "Guide: Creating the Definition of Done 231. Guide: Evolve the Definition of Done 240")

● At TGIF, everyone clearly understood that adding ‘fake’ team members (e.g. project managers, “PowerPoint architects”) to Scrum teams would not increase a team’s capacity or rate of output. Instead, it would just create an illusion of having a ‘bigger team’ (LeSS experiment: “Avoid… Fake team members 235”)

● Senior leadership was comfortable, to be referred to as “IRS” (Impediments Removal Service), not as a “change management group” (LeSS experiment: “Try… Impediments service rather than change management 381”)

● Multisite planning poker (estimation poker) was already effectively used, by leveraging some reliable in-house built light electronic tools and video equipment (LeSS experiment: "Try… Seeing is believing—ubiquitous cheap video technology and video culture 425. Try… Multisite planning poker (estimation poker) 429")

● Centralized (organizational) coaching departments and agile/lean cookbooks - were already viewed as a sign of faking/rebranding and local optimization that would eventually lead to building ‘ivory towers/organizational silos of privileged coaches (LeSS experiment: "Try… Prefer decentralization solutions over centralization ones 206. Try… Central coaching group 399. Avoid… Central coaching group with formal authority 399")

Learning Lessons from Adopting Basic Scrum

Prior to LeSS adoption effort, TGIF had a predominantly componentized structure. There were soft-line boundaries between the subgroups of business analysts, manual testers, developers and technical leads/managers. Requirements intake was typically done by tech-side business analysts, interfacing with biz-side business analysts, with the former then molding collected information into business requirement documents (BRDs) or similar types of contractual documents, and then passing documentation further along, to developers and testers. The cartoon below describes this classic challenge:
This was a classic manifestation of a waterfall process, between siloed subgroups of individuals, belonging to the same organization and the same technology tower. The biggest challenges with this organization design came in the form of too many handovers between subgroups, finger pointing and blame-shifting for having a lack of clarity, high volume of errors and omissions, causing long lead time/cycle time, continuous task-switching and work “aging” (due to task switching and external hold-ups).

The initial attempt to implement basic Scrum (as mentioned above, prior to LeSS adoption attempt), came in the form of restructuring teams, by bringing together individuals of complementary skill set - together. However, each team member was still l-shaped, possessing only primary functional expertise and a technical skill. There was not much interest in learning new functional domains or technologies, as everyone's job title was written, as per classic HR definition of a role: ‘business analyst’, ‘tester’, ‘developer’,
Lack of motivation by people to learn new functional and technical skills could be easily traced to individual goal-setting and performance reviews, by first-line managers, that did not encourage anyone to expand their horizons and learn new things. Everyone’s end-of-year performance and discretionary incentives (bonuses, promotions) were based on an individual’s ability to prove that they delivered, and over-delivered, as per their original job descriptions (and did better than their peers/colleagues). Everyone was very risk-averse, and people did not want to experiment anything new and innovative (e.g. slowing down their primary work and learning a new skill set, or experimenting for the benefit of a team).

Among many other challenges that this mini-waterfall-intra-team design had caused, was inaccuracy in capacity management and inability to work in the order of business priority.

Below is a graphic illustration of individual capacity management attempts that each team in TGIF group had to face almost every sprint (names and skills are substituted with fictional ones), at the beginning of their sprinting journey (they used 2-week sprint cadence).

Case 1 and Case 2 below - represent two independent attempts to account for individual level - and overall team-level capacity of one of the TGIF teams, by estimating along the dimensions of single-function specialty and single-technical skill set, respectively.

**Case 1: Impact of single-functional specialty of individuals on a team’s capacity management:**

<table>
<thead>
<tr>
<th>Team Member</th>
<th>Skill Set</th>
<th>Sprint Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>BA</td>
<td>60 hours</td>
</tr>
<tr>
<td>Jim</td>
<td>Manual QA</td>
<td>55 hours</td>
</tr>
<tr>
<td>Jeff</td>
<td>Developer</td>
<td>60 hours</td>
</tr>
<tr>
<td>July</td>
<td>Developer</td>
<td>50 hours</td>
</tr>
<tr>
<td>Jerry</td>
<td>Developer</td>
<td>65 hours</td>
</tr>
<tr>
<td>Josh</td>
<td>Developer</td>
<td>60 hours</td>
</tr>
<tr>
<td>Jill</td>
<td>Manual QA</td>
<td>55 hours</td>
</tr>
<tr>
<td>Total Team Capacity</td>
<td></td>
<td>405 hours</td>
</tr>
</tbody>
</table>

**Case 2: Impact of single-technical skill set of individuals on a team’s capacity management:**

<table>
<thead>
<tr>
<th>Team Member</th>
<th>Primary Skill Set Only</th>
<th>Sprint Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jeff</td>
<td>Java Front</td>
<td>60 hours</td>
</tr>
<tr>
<td>July</td>
<td>Java Back</td>
<td>50 hours</td>
</tr>
<tr>
<td>Jerry</td>
<td>SQL</td>
<td>65 hours</td>
</tr>
<tr>
<td>Josh</td>
<td>UI/UX</td>
<td>60 hours</td>
</tr>
<tr>
<td>Jill</td>
<td>Oracle</td>
<td>55 hours</td>
</tr>
<tr>
<td>Total Dev-only Capacity</td>
<td></td>
<td>290 hours</td>
</tr>
</tbody>
</table>

**Case 1** - With this team composition, with each team member having a specific (single) functional role, there was a much higher risk of a single point of failure that would lead to a hard block for the whole team. For example, losing BA for a sprint (illness or vacation) meant that the team would not be able to perform any analysis of forecasted work in that particular sprint.

**Case 2** - Almost equally challenging, with this technical skill set distribution among developers, with each developer possessing only primary, but no secondary skill, losing SQL person would put in jeopardy work on an entire application component: database in that particular sprint..

While teams tried to do their best at preserving democracy (no pecking order) and making team commitments (not individual), effectiveness of planning sprint work, with so many internal limitations and risk factors, was clearly compromised. Pulling work items from a
backlog, in the order of priority, was close to impossible, because capacity limits of a particular functional expertise or technical skill set were be quickly exceeded. For example, if PBI # 2 from the top required front-end Java work but a team (more specifically, only Jeff) already used up all of his capacity on PBI # 1, a team would have to skip PBI # 2 and go to an item of lower priority that did not require any work on a back-end component, using Java. But even more problematically, such single-functional specialty team composition, would lead to many PBIs becoming a representation of work of a person with a particular skill set. For example, having an abundance of ‘analysis-only’ stories, created by and for consumption of BAs-only (John), was manifestation of a classic resource management and projectized resource planning, done by the manager BA group. This led to local optimization in a backlog and kept BAs busy, producing analysis stories in high volume. Similar cases were observed with architects, producing architecture stories-only and testers - producing testing stories-only, etc. And while everyone seemed to be overly busy with their own work, delivery of business-centric, cross-component-cutting PBIs remained low.

Although multiple TGIF teams had been experimenting with basic Scrum for some time, their efforts were aligned to application components, not cross-component-cutting features. As a result, they were constantly facing cross-team integration problems that required ‘hardening/release’ sprints, after every few sprints of component-centric development. Furthermore, it was practically impossible to find a real Product Owner to accept work, done by a single application (or system component) team - because no single application/component represented a customer-centric dimension. It was clearly understood by everyone, including business counterparts of TGIF, that component-centric team alignment, could only produce a deliverable in the form of ‘IOU’ (“I owe you”), not in the form of a shippable product feature that had a measurable, intrinsic business value.

To illustrate this, existing component releases routinely required several weeks of User Acceptance and System Integration Testing with “sign-off”, coming from third-parties, outside of TGIF space. Sprint interruptions were frequent, feedback cycle was long, and this led to significant waste through context switching. In addition to interruptions, coming from in-flight releases, teams were all too frequently encumbered with support for legacy applications.

One of the early concerns that was raised by developers, at the beginning of LeSS adoption journey, was about potential friction between frequently delivered potentially shippable work, expected of LeSS teams, and slow/cumbersome release processes by the rest of the organization. Specifically, the worry was around potential integration problems and continuity/automation of deployments. It was a valid concern. As the result of this thinking, it has become one of the key goals of LeSS adoption - to define/expand
product definition to such an extent that there would be minimal dependency on other systems, applications and processes. More about it below.

Defining (and Expanding) Product Definition

(LeSS principle: “Whole-product focus—One Product Backlog, one Product Owner, one potentially shippable product increment, one Sprint—regardless if there are 3 or 33 teams. Customers want the product, not a part.”)

The business domain, that TGIF and BGIF people worked on was called “Document Management” - the global system that was intensely used by BGIF people, for intake, transmission and storage of legal and financial documentation (electronically). Initially, prior to LeSS adoption, this domain consisted of a few narrowly defined so-called products, none of which were composed of complete cross-application/component-cutting, customer-centric features. The so-called products were defined, based on grouping together of various internal applications that historically, were owned by the same person (usually Application Owners). Other words, organizational design and sphere of personal influence and control defined product segment composition. This resulted in the following ‘products’ definition:

- Documents Capture CAP - Greenfield. The process of capturing data from another upstream system, external data-feed or manual entry
- Documents COM - Greenfield. The process that was responsible for communicating captured data from system to system and, ultimately saving it as a document
- Documents management platform CAR - a legacy platform that stored all legal documents

While all of the above-mentioned ‘products’ were a part of a much bigger Document Management system that had real customers within BGIF community, finding real customers that would truly care about delivery for any of the above-mentioned disjointed product segments, with the latter being ‘wrapped’ into fake projects, was rather challenging. This is something that had to be corrected: (LeSS Experiments: Avoid… Fake team-level “Product Backlogs” 132 | Avoid… Projects in product development 238)

Prior to LeSS adoption, the original alignment of workers within TGIF was along the dimensions of segregated segments (sub-systems) of Document Management, as follows:
Realigning Teams (from Sub-Systems to One Product)

First, the coaches were presented with the dilemma of how to reorient development of the three product segment teams in a way that every team could work on each of the segments: front to back, independently, including their underlying applications. Based on the total number of people involved in the effort, their skill set and domain expertise, it was agreed that from three disjointed product-segment-centric teams (CAP, COM, CAR) it would be possible to produce three cross-product-segment teams (CapComCar Team 1, CapComCar Team 2, CapComCar Team 3) that would be able to work on all three product segments.

But before moving on with this effort, something else had to be done first: Senior Leadership had to be presented with an eye opener that segment/component-centric development was the main root cause of integration and excessive cross-team coordination problems (as in waterfall development) that required multiple ‘hardening/stabilization’ sprints - and, therefore, was costly and time consuming.

Why was senior leadership the first one to be approached with this dilemma?

It was important that they understood some important organizational implications of moving from components to features. Specifically, with feature-centric development,
usual domains of control/ownership, by component and application owners, would be loosened, in favor of feature teams, being allowed to touch multiple components, independently. This could potentially, create a lot of resistance and *turf protection* attempts, coming from former component owners/managers. Senior leadership needed to figure out a way to ensure that used-to-be component owners/managers did not feel deprived or depreciated, by the organization. This was done by offering component owners/managers an opportunity to act like component teachers/mentors, by offering guidance and learning experience to feature team members that were allowed to work with multiple components (LeSS Guide: Component Mentors 304).

Then, the **initial training workshop** took place.

It was held for all teams involved in the product-*segment* development, also including management and stakeholders (business and technology). The goal of the workshop was to identify ("carve out") what would better fit the definition of a real product, from a standpoint of Product Owner. It required understanding how different sub-components/applications of the above-mentioned product segments, would communicate with one another during *runtime*. (Note: sub-components/applications that were not a part of the above mentioned three segments were also identified but left out of further discussions. They are schematically shown colored in white, in the diagram below).
During this workshop, TGIF people, discussed what skill set and/or domain expertise were required by each team, in order to ‘cut across’ multiple sub-components/applications, of all three application segments. Based on this information and individual preferences, TGIF people self-organized into teams. Some line managers were on ‘standby’ to address any organizational/HR-related issues, should those arise (luckily, none did) but were explicitly asked to stay out of self-organization process (outside the room).  (LeSS Experiment: “Avoid… Single-function teams 155 • Avoid… Component teams 155 • Try… Feature teams 174”).

Initially, three (3) collocated teams were formed to work on the same Product Backlog. This number subsequently grew to 5, when everyone realized that in order to have a higher output, LeSS Sushi Roll needed to add a few more properly structured cross-functional teams (as opposed to further staffing existing teams with single-function specialists, and by doing so, violating the Brook’s Law).

First, the teams were additionally educated (structured training) on structure and purpose of all LeSS-specific joint events. This helped people understand what additional value LeSS-specific events have, on top of basic Scrum events.

Then, the following was implemented by the teams:

- Joint Sprint Planning, with whole teams in attendance (this has changed to having team representatives only, not whole teams, when the number of team has grown to five - in order to minimize the level of noise and disengagement) was introduced.
- Corollary to the above, Sprint Planning Part 1 and Part 2 became more clearly distinguished, in purpose, nature and attendance. While Part 1 had just representatives from the three mature, teams, plus all members from the newly added teams, Part 2 became team-specific, with some teams deciding to join their respective sessions, in situations, where they suspected cross-team dependencies.
- Overall Product Backlog Refinement (PBR) sessions - the teams followed the same approach, as with planning sessions, by splitting their Overall PBR and PBR (for each team)
- Joint Sprint Review - now attended by all teams, without exception, with Product Owner, users and stakeholders, coming along
- Overall Retrospective, in addition to Team-specific Retrospectives, having the former attended by Scrum Masters, Product Owners, and technical management of TGIF (the latter was invited to learn and take responsibility of removing organization-level impediments).
**Identifying Product Owner and SMEs**

In the **next workshop**, the coaches facilitated a persona-creation exercise: with real people that represented a spectrum of use-cases for each product segment, as well as for an entire product. It turned out that regardless of the conventional segmented approach in product development, the actual activities that product users had to perform spanned across multiple segments, for practically each of the following business activities:

- Client onboarding
- Credit risk
- Documentation management (TGIF)
- Legal
- Collateral operations
- Sales and traders
- Regulatory reporting

For example, the persona “Rita Regulations” was introduced, to better understand how an end user would interact with the system:
Below, are some additional roles that were identified during the exercise of user role modelling:

![User Profile Images]

(some additional relevant working artifacts):

Here is a real example of a brainstorming activity in the workshop, to help with better understanding of how Cap product segment worked:

“...it allowed all VBB documents to be ingested and categorized according to a common-model. The important implication from a legal and regulatory perspective is that the ingested documents could be indexed and eventually made searchable. The amount of metadata for documents spanning all business areas for a large financial organization was vast. The main technical challenge was to build a responsive data store that could meet the needs of the business. Since the data model was also continuously evolving, the system needed to be able to adapt without significant disruption or reprocessing of stored documents."

Similar brainstorming discussions were held around COM and CAR product segments.

Ultimately, the new <true> product vision was also formulated in the workshop (LeSS Experiment: Try... Add and do a cross-product common goal 128). The vision was no
longer just from a standpoint of CAP segment-only, or COM segment-only, or CAR segment-only. The vision now spanned across all three product segments (“CAPCOMCAR”), as it became clear to all that the latter three were tightly coupled and supported the same, unified business process.

Here is a real example of the vision statement for CAPCOMCAR:

“VBB employees and external clients will prefer to use our document capture platform as the most efficient solution that enables timely setup and management of products and accounts and conforms with global, legal, regulatory and control requirements. The process will include information capture, communication and storage”

During the workshop, both business and technology stakeholders, agreed on the high-level product themes and strategic, long-term goals. All future work was captured in one shared CAPCOMCAR Product Backlog (LeSS experiments: Try… Merged product backlog for a set of products 256 | Try… Focus on the overall product 193), in the form of PBIs of different size/complexity and priority. Each of the newly formed, teams was now able to take any work item from a backlog, regardless of a product segment it belonged to originally (LeSS experiment: Try… Team works on multiple products 257).

Once CAPCOMCAR Product was defined and initial product backlog has been agreed to, the teams were presented with another big dilemma: who would take on the key role of Product Owner and who would be selected from a large group of SMEs and stakeholders to become responsible for additional supportive work (e.g. clarifications).

Identifying individuals for the role of SMEs/stakeholders was relatively straightforward. The decision was made to pick out three individuals that had a lot of hands-on knowledge with each one of the above-mentioned product segments (steps in Document Management process): CAP, COM, CAR. For the most part, they all turned out to be the same individuals that used to support segment centric development teams in the past :).

On the other hand, identifying someone for the role of Product owner was much more challenging. It turned out that no one from the community of SMEs/stakeholders had good understanding of the overall CAPCOMCAR product scope and overall strategy. Everyone was too much focused just of their own segment of the product. At a given organizational level, it was practically impossible to find a person that would have a sufficient strategic vision and enough organizational empowerment to either have priorities or desire to engage with multiple teams. It became clear that the real Product Owner would have to come from a higher echelon of organizational structure.
Through a series of learning sessions and system modelling techniques (causal loop diagrams) that the coaches conducted with technology and business leadership, it was discovered that many pitfalls could be avoided if a real, empowered Product Owner was identified. BGIF side of LeSS construct impressed everyone by their learning and understanding, and after a short period of time a more senior business person was identified for the role of Product Owner.

Eventually, LeSS construct, from a standpoint of Product Owner and SME/stakeholders, support started to look like this:

![CapComCar product diagram](image)

**Expanding LeSS by Adding More Teams**

While partial flip of technology application-specific/component-centric teams (only three) into product-specific/feature-centric teams has produced initial fruitful results, the rest of TGIF group remained in its original state. Soon, it became apparent that in order to increase the rate of output from a CAPCOMCAR product delivery perspective, more bandwidth was required.

It is important to note that the need for higher output was wanted not because of a classing Contract Gaming that is frequently seen in organizations, where business asks for “more and more” and technology pushes back with “less and less”. On contrary, in this case the desire to have a higher demand was more in the form of “we [business] really like what/how you [technology] have been producing and we just would like more of it. What would it take to have more of your product delivered to us”? 
To honor the above described needs, using the composition pattern of the first three LeSS construct teams, that were already operational, two (2) more teams were created, by leveraging human assets (please, don't call intellectual workers as ‘resources’, it is not nice!) from the same technology space, and put through some initial preparatory steps, before being allowed to join the existing LeSS construct.

These preparatory steps included: a refresher training on basic scrum, introduction to LeSS (guides and experiments) with emphasis on joint events (as described above), familiarizing new teams with the backlog, priorities and strategic goals, and only then, finally “flipping a switch”, by bringing them into LeSS, with the next upcoming sprint.

One of the experienced Scrum Masters from the first three mature teams was then asked to pick up the new teams, whereas the other Scrum Master widened his focus across the mature three teams (now, the LeSS of five teams had two Scrum Masters).

A few most senior developers from the first three teams, were asked to temporarily take on the role of developers-travelers (LeSS experiment: Try… Travelers 207), and join the new teams, to lead by example and provide some guidance. The travelers, temporarily, became a pivot for each of the newly created teams, by teaching new members not just dynamics of LeSS but also certain nuances of different product segments that the ladder did not know. Having some people travel, temporarily (for a few sprints) lowered capacity of the original three teams but was considered as a worthy investment by everyone, because it helped expediting and getting up to speed new joiners. Furthermore, the travelers were able to seed engineering great-practices across the organization.

To speed up the process of assimilation with more experienced teams (in terms of product backlog knowledge, ability to estimate work, etc) all teams agreed that both new teams would be coming to joint LeSS events (Sprint Planning 1, joint PBR) in full, whereas the
three seasoned teams would be sending just representatives (to minimize the overall amount of people in meetings). Also, for a few initial sprints, the two new teams went into multi-team Sprint Planning 2, with at least one of the original three teams. This was done to improve learning experience/knowledge transfer of new joiners.

**Improving Overall Interaction with Business**

Even after CAPCOMCAR product was properly defined and Product Owner was elected, communication problems between TGIF and BGIF did not completely go away. Direct contact between business and technology still remained to be an issue. Mainly, this came in the form of TGIF team members confusing priorities of Product Owner (and business decisions, in general) with managerial decisions that came directly through management lines. This was because of still prevailing (outside of LeSS construct) original organizational setup, with managers requesting technical improvements from developers, by bypassing the established process of business approval, bringing requests directly to teams, and justifying them by perceived urgency.

Given the potentially severe implications to developers, of refusing to obey hierarchical lines, an experiment was attempted by the teams to inform the Product Owner of such managerial “emergency changes”. This was done implicitly, rather than explicitly, to avoid inflaming situations, and as follows:

On a few occasions, line management was invited in Product Backlog Refinement sessions, as guests, where they were asked, to state their most pressing needs explicitly and candidly, in front of Product Owner and some key stakeholders. This was discussed in the context of the teams’ capacity and priorities coming from the business. Very diplomatically, line management was put in a situation, when they had to negotiate not with the teams but with Product Owner directly - what priorities should be. The teams merely observed and contributed to a dialogue in various ways (e.g. clarifying capabilities, limitations, dependencies, etc). By removing themselves from potentially unsafe negotiations, the teams were able to focus more on work and less on politics. This approach has worked, as the teams were no longer as exposed and were out of harm’s way.

However, the systemic root causes of the “emergencies”, that were mainly caused by instability of old legacy systems and recurrent production problems, were still pretty difficult to isolate and contain.

Some examples of systemic root causes:

- Lack of test automation / low unit test coverage → bugs escaping to production → urgent production fixes required → unplanned work/interruptions coming in mid-
sprint → multitasking/task switching by teams → **negative impact on delivering planned work**

- Ineffective hiring policies (job descriptions, compensation schema) → problems with talent acquisition → lack of multi-skilled/T-shaped developers → scarcity of specific skill set when it is needed most → “borrowing” skill set from other feature teams (asking people to perform side-work, outside of Sprint and Product Backlogs - *more about in the section below*) → disrupting team’s work dynamic → **negative impact on delivering planned work**

**Minimizing Side-work and Supporting Single Product Backlog**

Due to a significant amount of pre-existing intra-sprint interruptions within TGIF group, it became critical to increase transparency on the associated dollar-cost of unplanned/"hidden"/side-work. As it is often the case within large enterprise organizations, facilitating big changes required an empirical approach. For this, the teams were advised to stop using a separate backlog to manage interruptions and side-work, and instead keep all of their work in one shared Product Backlog. This allowed Product Owner to refer to one single “source of truth”, instead of multiple independent “containers of wishes” - to see the overall amount of work the teams were asked to deliver, with conflicting priorities, and by doing so, set *real* priorities.

Taking into account inevitable work interruptions, helped managing the teams’ capacity and better reflect it during planning. After several sprints, senior leadership was presented with the data that illustrated how ad-hoc, unplanned/interruptive work diluted the teams’ attention and shifted their focus away from planned and forecasted product-centric work. This led to making some hard, but necessary, choices: to mandate from requesting individuals and business groups that tried to put their priorities on top of and around Product Owner’s priorities, to stop doing so. While in practice, LeSS teams still had to cope with some ad hoc, legacy technical work, at least now, they were able to clearly visualize this problem to Product Owner and senior leadership and manage their work more accurately, by not overstretching/over-utilizing their forecasts beyond capacity ([queue size vs. capacity utilization](#) ‘teaching’ was delivered, as it referred to in LeSS experiments: “Try… *Visual management 71, Try… Visual management to see the invisible queues 111*”
Providing a Non-hierarchy Based Escalation Mechanism

As mentioned above, the hierarchy of VBB was such that it led to command and control behaviors between reporting layers. Any action that, rightly or wrongly, deviated from the path dictated by management lines, or challenged existing norms and order, was often seen as risky and was discouraged.

Most attempts to put to light underlying problems and impediments went in vain. This was indicative of an endemic problem, where historically the company actively resisted to changes.

The coaches were presented with the following problem: how to take senior leadership admit that problems exist and require attention, and then make everyone else comfortable to step up, speak up and engage.

To overcome this barrier, a compromise solution was proposed. It provided, at least, a safety-net for some more courageous individuals, who would otherwise have had only the option of escalating through the hierarchy and putting themselves at risk, to get their observations and concerns brought to light and to senior management.

As an experiment, the coaches have conducted a few sessions with senior leadership, educating the latter on the benefits of gemba/go-see. (*Lean thinking—Create an organizational system whose foundation is managers-as-teachers [........] who practice Go See at gemba.*)

The leadership was encouraged to resort to various discrete escalation techniques that would encourage, individuals from lower organizational levels communicate directly to higher organizational levels. The “IRS” (Impediment Removal Service) was introduced, as means of safer escalation of problems from teams and Scrum Masters to senior leadership. It came in the form of a dedicated internal email address alias (*impediment_removal_service@VBB.com*) that anyone could submit their personal challenge/problem in the form of a community discussion. Senior leadership was strongly encouraged by the coaches to diligently attend to DL exchange and engage in discussing pressing issues.

Improving Engineering Practices

**Test**  - “Avoid… Test department 32 Avoid… Separate test automation team 37, Try… Zero tolerance on open defects 39 • Try… Acceptance test-driven development 42 • Avoid… Traditional requirement handoff 46”

Manual testing practices were well-established within VBB and this meant those proficient in testing were distributed in abundance across newly formed Teams. Test automation and sharing good practices was strongly encouraged across the teams within the
organisation. The established process of “testing a release candidate” with a manual test guidelines was re-oriented to testing features as part of the Definition of Done. (Note: Below, is the illustration of Definition of Ready/Done/Undone derived by all teams in one of the workshops):

Since teams were working on a greenfield project, there was a great opportunity to enforce that automated tests were written for every backlog item. Unfortunately manual testing remained in two cases, both at the request of the teams:

1. The first was Integration Testing performed by the team in environments closer to production.
2. The second was User Acceptance Testing performed by an external, separate group.

Both, the teams and technology/business managers, felt it was important to continue this practice until trust was built in the automated toolset. Initially, this resulted in significant ongoing overheads. Interruptions to the teams existed because of slow feedback and a plethora of release related sign-off meetings that made it very difficult to facilitate positive change across the board fast.

**Design & Architecture** - “Avoid… System engineers and architects outside of regular feature teams 300, Try… Technical leaders teach at workshops 302 • Try… Hire and strive to retain master-programmer ‘architects’ 302 • Avoid… Architecture astronauts (PowerPoint architects) 302”

Although perhaps being something to avoid, the TGIF technical tower had an architecture function external to the teams. Amongst other things, the architecture team was
responsible for defining the high-level multi-year landscape of products in the tower and how the latter would fit together in the broader context of the company. In practice the architecture group was able to find ways to effectively collaborate with Teams and work as an engaged stakeholder to identify when specific architectural tasks would be required for a given PBI in Product Backlog Refinement sessions. The architecture group allowed LeSS teams to fully own implementation, without enforcing any low-level details. Some of the most senior architecture group members started running CoP-like workshops to educate individual teams on VBB-wide mandates and standards that drove architectural decisions.

In the large-enterprise context of VBB, the architecture group was also able to provide “air-cover” for LeSS teams in the technology vertical and even helped promote new product features.

There were frequent attempts from external firmwide architecture groups that had the potential to be very disruptive, to cause confusion and delays for LeSS teams, if TGIF organization architecture team did not intervene and protect. Still, the TGIF architecture group, unfortunately, did not completely dissolve and fall into individual Teams; they still remained as “architecture jet fighters” (they were closer to action) and were protecting TGIF LeSS teams from “architecture space aliens” (they were too far away from action).

In addition to architecture groups, technical management lines also wished to dictate architectural directions, from time to time. This was certainly the case in TGIF, where a more senior manager would introduce (without consultation with the teams) a weekly “Design meeting”. This had the effect of severely hampering work in-flight and future work, and was a source of conflict for the senior developers across all teams. Due to the seniority of the meeting orchestrator, this was extremely difficult to course-correct. It took a number of sprints until the problem was recognized by senior management and this low-level centralized decision-making attempt was abolished in favor of joint LeSS events.

**Legacy Code** - Try… “Increase organizational support for learning development skills 340  Try… Support more self-study 341  Avoid… Trivializing programming 341”

Across the firm, VBB was implementing several initiatives to provide metrics on code quality. This meant that there was a baseline “score-card” for key source code metrics. Given the size of the company and the sheer number of lines of code, the baseline scorecard had become too relaxed to be useful. An experiment was therefore conducted, specifically within TGIF, to introduce static source code analysis with the latest available rulesets across all languages within the group. Through regular on-demand scans, teams were encouraged to monitor changes to the code quality and PBIs would in some cases not be accepted based on the results. Monitoring of metrics including the following complexity; code coverage; duplications and security vulnerabilities etc. all formed part of
the Definition of Done. More in-depth reviews were carried out on an ad-hoc basis as required. With code quality becoming a regular part of the development process, standards improved.

**Forming Communities of Practice**

*(LeSS Experiment: Try… Cultivate Communities of Practice 252 • Try… Use CoPs for functional learning 253.)* Several Communities of Practice were also seeded during the initial phase of a coaching engagement.

This started with the **Scrum Master CoP**. For example, in Scrum Master CoP, a lot of emphasis was made on crafting facilitation skills of SMs and their ability to distinguish between team-level (internal) impediments and organizational (external) impediments. Based on the community involvement/participation it was also easy to differentiate between individuals that could become great potential candidates for the role of Scrum Master and those that would be not suitable for the role because of their mind set, behavior and career goals ("left-over" people). Through the community, Scrum Masters were able to further develop a mindset of coaches, enablers and enablers. Most experienced Scrum Masters exchanged with one another their facilitation tactics and other special techniques.

Another common theme across all of TGIF teams was a lack of maturity in automated testing. To remediate this, **Test Automation CoP** was created. This allowed experience to be shared across teams in a safe way - to agree and establish better testing practices. Noteworthy, that before adopting LeSS, TGIF had a soft boundary that separated dedicated test automation specialists from the rest of developers. This was not a formal organization separation, but it did create an element of a silo, segregation of duties and intra-TGIF handovers. By shifting from "softly defined" dedicated test-automation groups to Test Automation CoPs, most senior test automation experts were now encouraged to become less of ‘owners of automation process’ and more of CoP leaders and mentors, who would teach the techniques to others. This gave birth to gradual reabsorption of test automation activities into respective teams. Subsequently, each team’s Definition of Done became more mature, as test automation was added to it. Respectively, this alleviated the need of having ‘undone’ (test automation) work, to be handled by dedicated teams, outside of feature teams/their sprints *(LeSS Experiment: Try… Eliminating the ‘Undone’ unit by eliminating ‘Undone’ work 245)*
“Narrowing the Gap between Science and Business” (Daniel Pink)

(LeSS Experiment: Try… De-emphasize incentives 270 • Avoid… Putting incentives on productivity measures 271 • Try… Team incentives instead of individual incentives 272 • Try… Team-based targets without rewards 273 • Avoid… Performance appraisals 273)

LeSS adoption at TGIF was an experiment of organizational restructuring, inside the enterprise that historically had a deeply rooted conventional, hierarchical, top-down command & control culture. Both, the coaches and senior leadership, understood that in order for LeSS to succeed, there was a need to adopt the concept of 'organization within an organization', by ring fencing LeSS adoption efforts with protective boundaries. Outside of those boundaries - the rest of VBB enterprise would remain ‘as is'; inside - LeSS Sushi Roll, composed of TGIF and BGIF, with minimal necessary management layers, customers and a other ‘instances’ of centralized organizational domains, such as finance, HR - coming in the form of locally controlled policies and norms.

Please note that term ‘instance’ is used here to signify the fact that it was possible at the time of the experiment to create de-novo, inside LeSS construct, the above-mentioned organizational domains, as independent organizational structures. However, what became possible, with support of senior leadership, was informal redefining of some norms, values and principles that were critical for LeSS success. For example, harm caused by employee stack-ranking, individual performance appraisals and monetary rewards (incentives, perks and bonuses) was recognized and partially addressed.

Specifically, senior leadership and mid-level management involved in the process of appraisals and rewards was educated on severity of harmful impact that the above-mentioned processes could make on individuals, teams and the organization overall.

As a part of leadership education, main themes and excerpts from some well-known publications were presented to parties involved. Some of the publications are below):
A few workshops were conducted for senior leaders and HR business partners to discuss:

- Research and studies of how individual performance appraisals and their linkage to monetary incentives can cause harm to employees and organizations (summarized, as case 1 and case 2)
- Systemic cause and effect relationships between individual performance appraisals/bonuses and health of Scrum dynamics. Below, is an example of a causal loop diagram that was used to visualize the problem:
Specifically, strong messages were delivered by the coaches to everyone that by extrinsic motivation (a.k.a. manipulation) that drives individual performance, also causes internal competition and antagonism among employees, and their willingness/lack of intrinsic motivation to work together in a mutually supportive way.

Since it was not possible to explicitly change enterprise-wide HR policies and norms (it would draw too much undesirable attention to LeSS adoption and would probably put it at risk), the decision was made to implicitly encourage TGIF workers to define their own individual career objectives and development plans in support of organizational agility. The management, now being educated, was more effective in spotting undesirable behaviors and antipatterns.

To address the above organizational constraints, the management tried to create environment conducive to improved behavior by developers. The following was made clear and continually stressed:

- Importance of becoming T-shaped workers
- Benefits of learning new development tools and techniques
- Behaving towards other teammates in supportive, team-like fashion, as well as valuing team performance, over individual performance.

Also, the language containing “I own”/“I delivered” was less encouraged than “We own”/“We delivered”. Team members were strongly encouraged to provide to one another more candid and genuine feedback during retrospectives to contribute to each other’s maturity and growth. Scrum Masters were also specifically coached on how to act during retrospectives, to prevent events becoming too inflaming and galvanizing. Above all, a very strong emphasis was made on customer happiness with the overall LeSS delivery, being the most influential factor that identified financial rewarding of teams, and subsequently, individuals.

While, at the end of the year, each TGIF employee was still delivered by his/her line manager a end-year review and “report card”, it had much less bearing on how an employee was rewarded financially. Individual adherence and genuine support of agile transformation efforts and LeSS adoption were valued more than individual delivery and heroics. Letter-grading was still assigned informally by managers (it was entered in a centralized system of record, as it was required by HR) but everyone understood and spoke freely about the fact that it had practically no value to anyone.
Introducing Agile Budgeting

(*LeSS Experiment: Try… Beyond budgeting* 261)

The idea of using dynamic forecasting (rolling-wave score-cards) was introduced to management and to people that were typically responsible to annual budget planning. It was done in the form of ‘comparing-contracting’ LeSS (Scrum by multiple teams) with “copy-paste scrum” (many teams doing their own, independent Scrum). A few agile budgeting workshops were delivered by the coaches to people that were involved in the process of budgeting (also, “Implementing Beyond Budgeting”, by Bjarte Bogsnes was discussed, and most important takeaways from this book were reviewed. People were also encouraged to subscribe to Beyond Budgeting Round Table (BBRT) newsletter, for on-going self-education and awareness.)

The following three main benefits of agile planning/budgeting for LeSS, were continuously stressed to management, finance people and other representatives from business. They were also the means of seeking additional organizational support for LeSS adoption:

- When up-to-five LeSS teams work synchronously, together (side-by-side), on the same widely-defined product (real), their shared understanding of work type, size and complexity (having certain scrum events together really helps!) is higher. As a result, when it comes to forecasting a completion of work (features), five LeSS teams would be able to estimate work more accurately than five loosely coupled teams (“copy-paste scrum”) that work completely independently, on unrelated initiatives. *Why would this be this the case?* Because in LeSS, through shared events, team representatives, would exchange information with one another about their own as well as their respective teams’ capabilities, skill set, domain expertise, as well as share knowledge and develop understanding and appreciation for each other's strengths, weaknesses, etc. In “copy-paste scrum”, where every Scrum team does its own unrelated work, this is not possible.

- Since all LeSS teams work for the same customer (Product Owner), there is a much higher chance that they could develop a shared understanding of product vision and strategy, as they would get it from the same authentic source (Product Owner) – this would lead to a more effective planning.

- Having more direct correlation between development efforts of LeSS teams (*output*, in the form of shared PSPI) and business impact (*outcome*, in the form of overall ROI), would make strategic decisions about funding much more thoughtful: customers would understand what business value they can receive, based on an investment they make. When real customers can directly sponsor product-centric development efforts, by getting real-time feedback from a marketplace and deciding on future strategy, they (customers) become much more interested in dynamic forecasting, as it allows them investing into what makes most sense, at the moment. Dynamic forecasting of LeSS, allows increasing/decreasing a
number of scrum teams involved in product development flexibly, by responding to increased/decreased market demands and/or product expansion/contraction.

**Flattening Overall Organization Design**

*(LeSS Experiment: Try… Keep the organization as flat as possible 241)*

Throughout LeSS adoption experiment, one of the key messages that was continuously delivered to senior leadership of both, TGIF and BGIF, by the coaches, was that organizational design is the first order (key) factor that is responsible for organizational culture, individual behaviors, norms, values, principles, tools, techniques. Various coaching tools and techniques were used for this: the above mentioned CLDs - to visualize system dynamics, confidential surveys, one-on-one coaching sessions with technology and business people involved in LeSS adoption - to explain individuals' perception, ambitions and motivation. Some cartoon-style graphics were used to convey the concept of ‘less is more’ and ‘bigger does not mean better’. (These graphics also became a part of the article published on less.works):

<table>
<thead>
<tr>
<th>Conventional organization</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LeSS organization</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image2.png" alt="Image" /></td>
</tr>
</tbody>
</table>
In sum, all techniques revealed the same thing: organizational complexity, thickness of management layers and the persistence of no-value-adding organizational silos, diseases organizational efficiently, happiness of workers and clients and, ultimately, adversely impacts organizational economics.

As a result, some steps were taken to improve dynamics within TGIF part of LeSS. They were as follows:

- Strong messages were delivered by senior leadership that becoming a ‘line manager’ (by title) and acquiring reportees did not equate to a guarantee of promotion or salary increase
- Number of people, reporting to the same manager, was maximized, as long as it did not violate some hard-written, company-wide HR rules (about # of reportees per manager). This was done to reduce unnecessary reporting layers
- Individuals were strongly encouraged not to perceive their immediate management as an impermeable layer, above and beyond which they were forbidden to go. On contrary, a series of skip-one-level events, and one-on-one with line-manager+1 touch points were instituted to increase visibility of team-level dynamics and to bring more senior management closer to real action: gemba/go-see (LeSS Guide: Management: Go See 125)
- To strengthen the message that the company’s directory reporting lines were not the most important determining factor of human communication, senior leadership (at recommendation of agile coaches) had instituted the Impediments Removal Service (“IRS”), as means by which, any team member (or non-managing employee) could escalate his/her problem or concern to senior leadership, without fear of repercussion.

Below, is the graphic illustration that was used to educate the organization on the concept of organizational flattening that was required for scaling organizational agility (adaptiveness) and maturing the organization from waterfall to LeSS:
Conclusion

LeSS adoption effort had a positive impact on the following fronts:

- Business customers were pleased with continuous delivery of business value. Moving from component/application-centric delivery to feature-centric delivery, drew interest of many stakeholders and end-clients. Also, the level of transparency and predictability brought by LeSS development has grown much higher to everyone’s satisfaction. Relationships between technology (TGIF) and business (BGIF) were at their peak best during the LeSS adoption experiment.

- On both sides, TGIF and BGIF, people learned how to identify organizational impediments and relate superficial discoveries to deeper, systemic root causes, that affected an entire system. This learning will stay with people for a long time.

- The created Communities of Practice became an effective media for individual learning and many people started using them very effectively for personal career growth.

- Agile engineering practices/DevOps grew deeper roots into TGIF and beyond, positively affecting technology group, at large.
A handful of seasoned Scrum Masters was nurtured and some of them became so passionate about becoming change agents that they have decided to make this into a career journey, inside and outside of VBB.

The following challenges remained unresolved at core, by the time both coaches departed. Unfortunately, some of the challenges were not completely removed, and continued to resurface as the coaches were exiting:

- Component/application ownership did not completely mold into mentorship
- "Left-over" people were not effectively accommodated by other parts of the organization (other areas, cross-training) and continued to put up invisible resistance to LeSS adoption
- Self-management remained a challenge, as organizational reporting lines and HR policies' limitations still reminded of themselves
- Mid- and end- year performance reviews, followed by subjective bonuses, continued to affect individuals’ morale and enthusiasm, even though their importance was trivialized
- Fiscal budgeting continued to hinder exploratory development and research, as everyone remained risk-averse in terms of asking for additional funding.