Why did your company decide to migrate?

Role	How many years have you spent in your role?	Number of employees of your team	Number of employees of your organization	Organization's domain(s)	System Developed	When was the application first created? (yyyy)	When did your company decide to migrate to microservices? (yyyy)	How large is the application (number of microservices)?	How many major releases it had?	Answers	CODING	MOTIVATIONS	CHARACTERISTICS (possibly derived from ISO25000 serie)	METRICS_BEFORE	METRICS_AFTER
Manager & Developer	4	3	250	online browser games	online games	before 2005	2012	50-60	weekly	to improve the maintainance of the code, to improve deployment, to organize the people better, split up people	maintenance, deployment strategy, team distribution,		Maintainability, team_organization	maintainability, deployability, team_independence	maintainability, deployability, team_independence
RTE (SAFe)	3, trained 100 people		10000	digital banking	banking	1980	2013	> 1000	> 1000	to shield the backend from the frontend, separate UI from backend	modularization	modularity	Maintainability		
Developer	3	9	a lot	AI, search	ibm watson	2016		70-80, 20 components	weekly	recomposable modules, modular design	modularization	modularity	Maintainability, Performance_Efficiency, COST	patterns_adopted	patterns_adopted
Manager	+5 years	10	35	I.S.V.		1997 - 2003	2017	10-15	monthly	marginal cost of adding one feature (even a simple one) become very important (3-4 weeks and 2-3 people); bugs in production; need to refactor legacy, without starting from scratch; need to scale	modifiability, fault tolerance, refactoring (maintenance), scalability	modifiability, fault_tolerance, maintainability, scalability	Maintainability, Reliability, Performance_Efficiency	time_to_market, performance, #bugs, test_coverage, deployment	Maintenance_effort
Scientist Researcher	<1	40	>100	translation, online, b2b	translation product, AI	2014	2017	> 10	roughly about 5-10	maintainance, to scale (people), were necessary because team size increased, to scale development with more employees, codebase too large	maintenance, scalability, large codebase, team organization	maintainability, scalability, large_codebase	Maintainability, Performance_Efficiency		
Agile Coach (independent)	5	8	1000+	telecommunication	consulting	2012		no approximation available, at the moment monolith	8	(does not know)					
Agile Coach	> 4		>250	banking	SIDAS, banking	2009	2017	> 10	biweekly	autonomy of teams, autonomy of deployment, autonomy of responsibility, time-to-market	team independence,deployment, [NO team organization], time-to-market	team_organization, deployability, time_to_market		performance, complexity	performance, complexity
QS	QS 3-4, developer 15	18	7000	tax, legal, hosting, b2b, b2c	tax consultant	2016	2017	> 100	not yet released	agile, adopt, market demands, work in progress, cost	AGILITY, DEPLOYMENT, [NO market demands], costs	agility, deployability, cost			
Manager	4	11	>300	e-commerce, logistics	e-commerce, logistics	2013	2015	7	>10	performance issues, maintainability purposes, team organization, fault tolerance/downtime issues	performance, maintenance, team organization, fault tolerance	performance, maintainability, team_organization, fault_tolerance		data management	
Analyst/Architecture	7	20	50	IT	production managment	2016	2017	10	2	cost, availability in the cloud	costs	cost, availability		#microservices, #requests	data complexity, #responsibility_per_service, available_devops_infrastructure, team_independence
Developer	4	40	~6500	public services	human resource application, process management, user access control	1992	2014	+800	we do not have this number, because systems are too old	because there were services doing the same thing spread for all system	modularity, [not sure reusability, refactoring]	modularity, reusability		duplications	
Developer and KDL	10	6	100	automotive	all	2016	we use microservices from the start	5	3	we use microservices for their reusability and to try to keep things simple (single responsibility)	reusability, modularization, development easiness	reusability, modularity, modifiability		response time, complexity, testability, service_responsibilities	response time, complexity, testability, service_responsibilities
Manager	12	8	150	car manufacturer	all	2016	(use microservices from the start)	111	annuely	mainly due to the share of resources of aour main products. allows to reuse a microservice in several contents	resource management, reusability	resource_utilization, reusability		maintenability	maintenability
Developer	8	6	>150	Software Development	Document Management System	2003	2016	8	15	Better separate each features, reduce maintenance effort, reduce synchronisation between teams	modularization,effort, [may be - maintenance], team organization, team independence	modularity, development_effort, team_organization		maintenance_effort, #bugs, #effort_per_bug, Velocity	maintenance_effort, #bugs, #effort_per_bug, Velocity
Manager	10	6	18	web development	Web cms to support users in automatically create websites (similar to wordpress.com)	2015	2017	6	2	To decrease coupling and because we believed that this is the future.	modularization, willingness	modularity, willingness			
Developer	10	12	>1000	Insurance	insurance information system	1986	2016	67	don't know, at least 3 in the last 10 years	Because we were not able to maintain the system anymore (Cobol on DB2)	legacy system, maintenance	legacy, maintainability			
customer site	na			finance	banking	1990		100+		development team, easier automated/isolated testing, continuous development	team organization, testability, deployment/delivery	team_organization, testability, deployability		Velocity, maintenance cost, #bugs	Velocity, maintenance cost, #bugs
Upper Manager	7	200	25000	bio-pharma	many large knowledge management systems	2015	no decision	100	40 sprints	the old system was written in Ruby On Rails and the knowledge to maintain the monolith got somewhat lost.	legacy system, maintenance	legacy, maintainability		modifiability, team_independence, technology independence	
Manager	4	3	30	research	research management system (RMS)	2013	2017	8	1					response time, #requirements	velocity, modifiability
Developer	5					2014	2017	80	8	complexity, # bugs	complexity, fault tolerance	complexity, fault_tolerance		complexity, #bugs	complexity, #bugs, scalabilty
Manager	6					2015	2017	100	6	maintenance process	maintenance	maintainability		LOC, change frequency	scalability
Developer	5					2015	2017	50	3	maintenance effort higher	maintenance	maintainability		Maintenance_effort	Maintenance_effort
Developer	6				e-coomerce	2013	2016	100	8	continuous development	deployment/delivery	deployability		response time, cost	response time, cost
Developer	4					2016	2018	50	2	Devops support	devops	deployability		infrastructure costs, maintenance_effort	infrastructure costs, maintenance_effort
Manager	7	20	100	Banking		2014	2017	40	6	maintainability, manager decision	maintenance, managerial decision	maintainability, willingness		#bugs	cost
Developer	5	10	150	e-commerce		2013	2015	90	8	team delegation of the responsibility	team organization, delegation of responsibilities	team_organization		velocity , scalability	velocity
Developer	4					2014	2016	50	4	Memory consumption problem , downtime of the system	HW resources optimization, downtime [No fault tolerance, scalability]	resource_utilization, availability		downtime, complexity, performance	downtime, complexity, performance
Developer	4	13				2015	2017	40	3	technology experimentation, delegation responsibility among the teams	team organization, technology experimentation, delegation of responsibilities	team_organization, technology_experimentation, team_organization		Modifiability, #bugs	Modifiability, #bugs
Manager	8		50	banking	banking	2014	2017	50	6	Maintainability effort, Devops support	maintenance effort, devops	maintainability, cost, deployability		complexity, #bugs, maintenance_effort, velocity, LOC	complexity, #bugs, maintenance_effort, velocity, LOC
Developer	10			IT consultant	IT consulting	2015	2017	50	3	manager decision	managerial decision	willingness		coupling, scalability, velocity	coupling, scalability, velocity
Developer	3					2012	2016	60	10	scalability, faultiness	scalability, fault tolerance	scalability, fault_tolerance		maintenance cost, #bugs, maintenance_effort, #effort_per_bug	maintenance cost, #effort_per_bug
Developer	12			IT consultant		2010	2016	90	10	coordination problem, team delegation	team organization, delegation of responsibilities	team_organization		maintenance_effort, #requirements, technology_independence	#bugs
Developer	8					2011	2015	100	8	Devops support, scalability	devops, scalability	deployability, scalability		scalability	scalability
Developer	10					2005	2015	95	20	delegation of responsibility, the majority of the new features to be implemented required more effort in the monolithic system	team organization, development effort	team_organization, cost		Modifiability, performance, velocity	Modifiability, performance, velocity
Developer	7					2009	2017	35	9	maintenance effort, automation process	maintenance effort, deployment/delivery	cost, deployability, maintainability			
Manager	5					2006	2016	70	10	too many bugs, maintenance process too complex	fault tolerance, maintenance complexity	fault_tolerance, maintainability		#bugs, LOC, complexity, velocity	#bugs, LOC, complexity, velocity
Developer	6	25	200	software house	digital marketing, sales	2007	2017	30	8	because we had no other choices	managerial decision	willingness		response time, #bugs, memory consumption	response time, #bugs, memory consumption
Developer	8	15	100	IT consulting		2011	2016	40	6	reduce overall system complexity	complexity	complexity		complexity, #bugs	complexity, #bugs
Developer	15	30	200	sw house		2009	2015	100	na	speed-up delivery time	deployment/delivery	deployability		response time, memory, maintenance_effort	response time, memory, maintenance_effort
Developer	8	25		e-commerce		2007	2015	50	10	maintenance became too complex	maintenance complexity	maintainability		change frequency, maintainability, maintenance_effort, #bugs	change frequency, maintainability, maintenance_effort, #bugs
Developer	10	20	100	sw house		2005	2017	100	15	reduce overall system complexity and maintenance effort	maintenance complexity, maintenance effort	maintainability, cost		maintenance effort, complexity	maintenance effort, complexity
Developer	5	20	100	IT consultant		2010	2017	50	9	speed-up delivery time	deployment/delivery	deployability		deployment_time, velocity, change frequency	deployment_time, velocity, change frequency
Developer	7	15	100	IT consultant		2008	2016	40	10	because we had no other choices (manager decision)	managerial decision	willingness		maintenance_effort, #bugs	maintenance_effort, #bugs, scalability
Developer	4	8	50	IT Consultant		2011	2017	30	7	reduce overall system complexity	maintenance complexity	maintainability		complexity, #bugs, modifiability	complexity, #bugs, modifiability
Developer	5	10	50	sw house		2013	2017	40	5	reduce maintenance effort and complexity	maintenance effort, complexity	maintainability, cost, complexity		maintenance effort, complexity	maintenance effort, complexity
Manager	10	30	200	It consultant		2004	2015	100	15	because we had no other choices	managerial decision	willingness		maintenance_effort, bugs	maintenance_effort, bugs
Developer	4	15		It consultant		2012	2015	30	5	- maintenance became too complex - speed-up delivery time - reduce overall system complexity	maintenance complexity, deployment/delivery, complexity	maintainability, cost, deployability, complexity		Maintenance effort, #bugs, velocity, complexity, #bugs	Maintenance effort, #bugs, velocity, complexity, #bugs
Developer	10					2007	2015	60	13	reduce overall system complexity	complexity	complexity		complexity, #bugs	complexity, #bugs
Manager	10	20	150	IT consultant		2011	2017	50	7	We have a huge codebase and a giant monolith that was extremely hard to manage	large codebase, maintenance	large_codebase, maintainability		response time, downtime	response time, downtime
Developer	10	6	30	IT consultant	b2b ecommerce platform	2001	2015	24	maybe 5 since microservices, we don't number	Being able to scale organizationally and technically	scalability, team organization	scalability, team_organization		team_independency, impact of failures, number of deployments,	team_independency, impact of failures, number of deployments,
Architect	10	250	650	E-Commerce, logistics	E-Commerce	2013	2015	150	Continuous delivery	Ease of development	Maintenance easiness/complexity	maintainability		cost	cost
Upper Manager	5	10	10	Fintech	Trading	2017	2017	7	1						

Perceived usefulness of metrics

Role	Scalability/Performance [RESPONSE TIME (The time between sending a request and receiving the corresponding response)]	Scalability/Performance [CPU UTILIZATION (The percentage of time the CPU is not idle)]	Scalability/Performance [PATH LENGTH (The number of CPU instructions to process a client request)]	Scalability/Performance [WAITING TIME (The time a service request spends in a waiting queue before it get processed)]	Scalability/Performance [IMPACT OF PROGRAMMING LANGUAGE (Communication between microservices are network based)]	Scalability/Performance [USAGE OF CONTAINERS (The usage of containers can influence the performance, since they need additional computational time compared to monolithic applications deployed in a single container)]	Scalability/Performance [NUMBER OF FEATURES PER MICROSERVICES]	Scalability/Performance [NUMBER OF REQUESTS PER MINUTE OR SECOND (Also referred as throughput or average latency)]	Availability [DOWNTIME]	Availability [MEAN TIME TO RECOVER (The mean time it takes to repair a failure and return back to operations)]	Availability [MEAN TIME TO FAILURE (The mean time till the first failure)]	Maintenance [TESTABILITY]	Maintenance [COMPLEXITY]	Process related benefits [DEVELOPMENT INDEPENDENCE BETWEEN TEAMS (The migration from a monolithic architecture to a microservice oriented changes the way in which the development team is organized)]	Process related benefits [CONTINUOUS DELIVERY]	Process related benefits [REUSABILITY]	Personnel Cost [DEVELOPMENT COST]	Infrastructure Cost [COST PER HOUR]	Infrastructure Cost [COST PER MILLION OF REQUESTS]
Manager & Developer	very/a lot	absolutely not	absolutely not	absolutely not	more than enough	absolutely not	just enough	absolutely not	absolutely not	absolutely not	absolutely not		very/a lot	absolutely	absolutely	absolutely	absolutely	absolutely not	absolutely not
RTE (SAFe)	very/a lot	more than enough		very/a lot	very/a lot			absolutely	absolutely	absolutely	very/a lot	very/a lot	very/a lot	absolutely	absolutely	absolutely	more than enough		
Developer	just enough	absolutely	little	absolutely not	absolutely not	absolutely not	absolutely not	absolutely not	absolutely not	just enough	absolutely not	very/a lot	very/a lot	just enough	absolutely not	little	absolutely not	very/a lot	absolutely not
Manager	absolutely	absolutely		absolutely		little	absolutely	absolutely	absolutely	absolutely	absolutely	absolutely	absolutely	more than enough	absolutely	very/a lot	more than enough		very/a lot
Scientist Researcher	absolutely	just enough	just enough	absolutely	little	very/a lot	just enough	absolutely	absolutely	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	absolutely	very/a lot	very/a lot	more than enough	more than enough
Agile Coach (independent)	very/a lot	little	little	just enough	little	just enough	just enough	more than enough	very/a lot	absolutely	very/a lot	more than enough	more than enough	very/a lot	absolutely	very/a lot	just enough	just enough	just enough
Agile Coach	very/a lot	just enough	just enough	absolutely	very/a lot	very/a lot	absolutely	very/a lot	absolutely	absolutely	very/a lot	very/a lot	very/a lot	absolutely	absolutely	very/a lot	more than enough	more than enough	more than enough
QS	very/a lot	more than enough	more than enough	very/a lot	absolutely	very/a lot	little	more than enough	very/a lot	very/a lot	very/a lot	absolutely	more than enough	very/a lot	absolutely	just enough	just enough	very/a lot	more than enough
Manager	more than enough	little	little	very/a lot	more than enough	more than enough	very/a lot	absolutely	absolutely	absolutely	absolutely	very/a lot	very/a lot	very/a lot	more than enough	more than enough	more than enough	just enough	just enough
Analyst/Architecture	very/a lot	very/a lot	very/a lot	absolutely	little	more than enough	more than enough	very/a lot	absolutely	absolutely	absolutely	absolutely	absolutely	very/a lot	absolutely	very/a lot	more than enough	absolutely	absolutely
Developer	absolutely	very/a lot	very/a lot	absolutely	just enough	absolutely	very/a lot	very/a lot	very/a lot	very/a lot		just enough	more than enough	little	more than enough	absolutely	little	just enough	just enough
Developer and KDL	more than enough	more than enough	more than enough	absolutely	very/a lot	more than enough	little	absolutely	absolutely	absolutely	absolutely	more than enough	very/a lot	absolutely	very/a lot		very/a lot	just enough	just enough
Manager	just enough	just enough	very/a lot	very/a lot	just enough	absolutely	absolutely	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	absolutely	absolutely	very/a lot	absolutely	very/a lot	very/a lot	absolutely
Developer	little	absolutely not	absolutely not	absolutely not	more than enough	absolutely	more than enough	more than enough	more than enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	more than enough
Manager	very/a lot	more than enough	absolutely not	absolutely	more than enough	absolutely	absolutely	just enough	absolutely	absolutely	absolutely	absolutely	more than enough	absolutely	absolutely	absolutely	absolutely	very/a lot	very/a lot
Developer	very/a lot	very/a lot	just enough	absolutely	very/a lot	absolutely	absolutely	absolutely	absolutely	absolutely	absolutely	absolutely	very/a lot	absolutely	absolutely	absolutely	absolutely	very/a lot	very/a lot
customer site	absolutely not	absolutely not	absolutely not	absolutely not	just enough	little	little	absolutely not	more than enough	very/a lot	little	absolutely	absolutely	absolutely	absolutely	just enough	more than enough	just enough	little
Upper Manager	absolutely not	absolutely not	very/a lot	more than enough	more than enough	more than enough	very/a lot	absolutely not	just enough	little	just enough	very/a lot	absolutely	absolutely	more than enough	just enough	just enough	absolutely not	absolutely not
Manager	little	absolutely not	little	little	just enough	absolutely	very/a lot	absolutely	just enough	more than enough	just enough	more than enough	more than enough	very/a lot	more than enough	little	just enough	more than enough	more than enough
Developer	little	little	just enough	just enough	more than enough	very/a lot	very/a lot	very/a lot	more than enough	more than enough	more than enough	very/a lot	very/a lot	more than enough	very/a lot	just enough	just enough	more than enough	more than enough
Manager	just enough	little	little	little	just enough	very/a lot	very/a lot	very/a lot	more than enough	very/a lot	just enough	very/a lot	very/a lot	very/a lot	more than enough	just enough	just enough	more than enough	more than enough
Developer	little	absolutely not	absolutely not	little	just enough	very/a lot	absolutely	absolutely	more than enough	more than enough	just enough	very/a lot	very/a lot	very/a lot	more than enough	more than enough	just enough	just enough	just enough
Developer	just enough	just enough	little	just enough	just enough	very/a lot	very/a lot	more than enough	more than enough	very/a lot	very/a lot	absolutely	absolutely	very/a lot	more than enough	very/a lot	more than enough	more than enough	more than enough
Developer	little	just enough	just enough	just enough	more than enough	very/a lot	more than enough	very/a lot	very/a lot	very/a lot	more than enough	very/a lot	absolutely	very/a lot	more than enough	very/a lot	more than enough	more than enough	just enough
Manager	little	little	just enough	more than enough	more than enough	more than enough	more than enough	very/a lot	just enough	more than enough	more than enough			very/a lot	very/a lot	very/a lot	more than enough	very/a lot	very/a lot
Developer	little	little	just enough	little	just enough	more than enough	more than enough	more than enough	more than enough	absolutely	just enough	very/a lot	absolutely	very/a lot	very/a lot	more than enough	very/a lot	very/a lot	very/a lot
Developer	little	just enough	little	just enough	just enough	very/a lot	very/a lot	very/a lot	very/a lot	more than enough	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	more than enough	very/a lot
Developer	just enough	more than enough	more than enough	just enough	more than enough	just enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot				more than enough	more than enough	very/a lot
Manager	just enough	just enough	little	just enough	more than enough	little	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	very/a lot	absolutely	absolutely	very/a lot	very/a lot	more than enough	very/a lot
Developer	just enough	just enough	little	just enough	just enough	just enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	absolutely	very/a lot	very/a lot	very/a lot	more than enough	more than enough	more than enough
Developer	little	little	just enough	just enough	more than enough	more than enough	just enough	more than enough	very/a lot	more than enough	more than enough	more than enough	very/a lot	more than enough	very/a lot	more than enough	more than enough	more than enough	more than enough
Developer	just enough	just enough	just enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	absolutely	very/a lot	absolutely	absolutely	more than enough	just enough	more than enough
Developer	little	just enough	just enough	just enough	very/a lot	just enough	very/a lot	very/a lot	very/a lot	more than enough	very/a lot	very/a lot	very/a lot	more than enough	more than enough	very/a lot	more than enough	just enough	just enough
Developer	little	just enough	just enough		very/a lot	just enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	very/a lot	very/a lot	very/a lot	more than enough	just enough	more than enough
Developer	just enough	little	just enough	more than enough	more than enough	more than enough	just enough	very/a lot	more than enough	more than enough	more than enough	very/a lot	very/a lot	more than enough	more than enough	very/a lot	more than enough	more than enough	more than enough
Manager	just enough	little	just enough		more than enough	more than enough	more than enough	more than enough	more than enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	more than enough	more than enough	more than enough
Developer	just enough	just enough	more than enough	little	just enough	just enough	just enough	more than enough	more than enough	more than enough	very/a lot	absolutely	absolutely	very/a lot	very/a lot	very/a lot	more than enough	more than enough	more than enough
Developer	little	just enough	more than enough	just enough	more than enough	more than enough	more than enough	more than enough	very/a lot	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	more than enough	just enough	just enough	just enough
Developer	just enough	little	just enough	more than enough	just enough	more than enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	absolutely	absolutely	absolutely	absolutely	very/a lot	more than enough	more than enough	more than enough
Developer	just enough	just enough	just enough	more than enough	more than enough	more than enough	more than enough	more than enough		more than enough	very/a lot	more than enough	very/a lot	more than enough	very/a lot	more than enough	very/a lot	very/a lot	very/a lot
Developer	more than enough	more than enough	very/a lot	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	absolutely	absolutely	absolutely	very/a lot	absolutely	very/a lot	very/a lot	very/a lot	very/a lot
Developer	just enough	just enough	more than enough	just enough	more than enough	more than enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	absolutely	absolutely	very/a lot	very/a lot	very/a lot	very/a lot	more than enough	more than enough
Developer	just enough	just enough	more than enough	just enough	more than enough	more than enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	more than enough	very/a lot	very/a lot	just enough	just enough	just enough
Developer	just enough	more than enough	just enough	just enough	just enough	more than enough	very/a lot	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot
Developer	more than enough	more than enough	more than enough	very/a lot	very/a lot	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	absolutely	absolutely	absolutely	absolutely	just enough	just enough	just enough
Manager	very/a lot	very/a lot	just enough	more than enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	absolutely	absolutely	absolutely	absolutely	absolutely	just enough	just enough	just enough
Developer	little	little	just enough	just enough	just enough	more than enough	just enough	more than enough	more than enough	more than enough	more than enough	very/a lot	very/a lot	more than enough	very/a lot	very/a lot	more than enough	more than enough	more than enough
Developer	more than enough	more than enough		more than enough	just enough	just enough	more than enough	more than enough	very/a lot	very/a lot	absolutely	absolutely	absolutely	very/a lot	absolutely	very/a lot			
Manager	just enough	more than enough	just enough	just enough	more than enough	more than enough	more than enough	more than enough	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	very/a lot	just enough	just enough	just enough
Developer	absolutely	very/a lot	more than enough	very/a lot	very/a lot	more than enough	very/a lot		absolutely	absolutely	absolutely	absolutely	absolutely	little	more than enough	more than enough	more than enough	more than enough	more than enough
Architect	more than enough	little	absolutely not	just enough	just enough	little	more than enough			very/a lot	more than enough	very/a lot	very/a lot	absolutely	absolutely	little	little	little	little
Upper Manager	very/a lot	just enough	very/a lot	very/a lot	very/a lot	little	very/a lot	absolutely not	absolutely	absolutely	very/a lot	very/a lot	very/a lot	more than enough	more than enough	just enough	more than enough	more than enough	more than enough

Role	Which information/metrics were considered before and during migration? Which information/metrics do you consider useful?	Not_measured(Should had)	CODING	Is there any measure that is not easy to collect?	CODING	How useful would you consider a discussion of the previous information before migration?	Do you think the factors or measures support a reasoned choice of migrating or not?	How easy is the set of factors and measures to collect and use?	Is there any measure that is not easy to collect?	Would you use this set of factors and measures in the future, in case of migration of other systems to microservices? If not, please motivate.
Manager & Developer	because of the benefits mentioned before	complexity		maintainability, complexity	maintainability, complexity	Very / A lot	yes	easy	maintainability, complexity	yes
RTE (SAFe)	(did not know)	Consistency_development, complexity, modularity	consistency in development	complexity	complexity	Very / A lot	yes	easy but we haven't got all facts	complexity	(not sure, already migrated)
Developer	architectural patterns, technical methods/techniques	deployment_cost, complexity, team_independence, modularity	cost of deployment, reliability of deployment, monitoring	complexity (many influences on different areas, this might be too big of an idea as "complexity" alone), Independence between teams (you'll probably reorganize anyway...	complexity	More than enough	yes	cpu and utilization, cost -> easy to count, estimate, measure; reusability, complexity -> reasonable, proxy needed; seems like with the right logging you might measure many of these things	complexity (many influences on different areas, this might be too big of an idea as "complexity" alone), Independence between teams (you'll probably reorganize anyway...	yes
Manager	lead time (idea to customer); performance (req/sec, memory, cpu); bugs (reported by customer); code coverage; deployment	maintenance_effort				3	yes from a technical point of view, but in my opinion it lacks of business considerations			yes
Scientist Researcher	unknown	availability, data_management, maintenability, performance	data management	no, they should be easily be measurable fro a running system	availability	6	yes	not easy to get people together to collect the measure	no, they should be easily be measurable fro a running system	yes
Agile Coach (independent)	unknown	response_time, meantime_to_recover, testability, complexity		response time, meantime to recover, testability, complexity	response time, meantime to recover, testability, complexity	6	yes	missing currently in his projects; most easy; some critical metrics missing	response time, meantime to recover, testability, complexity	yes
Agile Coach	performance, complexity (interactions between microservices)	team_independence, deployment_time, time_to_market		complexity of monolith	complexity	Very / A lot	yes	not so easy	complexity of monolith	yes
QS		available_dev_ops_infrastructure, velocity, deployment, cost	monitoring	business -> more complicated, security -> complicated, legal -> very complicated	business, security, legal aspects	6	yes	technically -> easy	business -> more complicated, security -> complicated, legal -> very complicated	yes
Manager	roadmap (planning features), feature importance in terms of user useful, data safe/validity	performance, maintainability, team_independence, bugs		i would say that process related benefits are harder to meesure	process related metrics	Very / A lot	yes	Fairly easy. lot's of tools provide many of the referred info	i would say that process related benefits are harder to meesure	yes
Analyst/Architecture	number of microservices (estimated), number of requests from critical processes	availability	data management, #access to database	continuous delivery, team dependency	continuous delivery, team dependency	6	yes to have a supporting argument to perform a breaking change of architecture	easy, with exception to the process-related ones...	continuous delivery, team dependency	yes, all
Developer	we did not use a metric, we only verify parts of the systems that are duplicated, and we have migrated it to microservices	impact of programming language, testability, complexity, reusability, modularity		impact of programming language, testability, complexity, reusability		Very / A lot	yes, i do	depends on the metric	impact of programming language, testability, complexity, reusability	yes, i could use it in the future
Developer and KDL	repsonse time, code paths/complexity, service responsability/testability	reusability, modularity, modifiability		yes		6	yes	relatively easy. we have lot of information from cloud foundry/appdynamics and api consumers that help us understand who is using our api and how	yes	
Manager	maintainance that was the main trouble before migration.	deployment_cost, resource_utilization	maintenance, deployment	not easy to collect the initial amount of effort to do the migration	effort of migration	Very / A lot	yes. depends on the number of business contents. But ideally the main factor is the low level complexit and in consequence the low cost of development for maintenance	relatively easy. depends on how much the stakeholders committed to the migration. being the main factor the number of issues in monolithic	not easy to collect the initial amount of effort to do the migration	yes
Developer	effort per user story. Number of faults, effort to solve each fault. Number of user story implemented per sprint All of them useful, except #user stories per sprint .		effort per user story, #bugs, effort to solve fault, #user stories per sprint	effort data in general	effort	More than enough	yes	not very easy, some measures should be collected manually before migrating.	effort data in general	yes
Manager	none	maintenance_effort, effort_per_bug	maintenance, fault solving effort	cpu data, response time and benchmarks in general	cpu data, response time, benchmark data	Very / A lot	yes	easy	cpu data, response time and benchmarks in general	
Developer	None, we should have collected some but now is too late.	downtime, time_to_market, testability, scalability	issue management, downtime, time-to-market, testability, scalability	benchmark data.	benchmark data	Very / A lot	yes	not easy, we should set a system to collect them, and on our mainframe it would have been very expensive.	benchmark data.	may be, depending on the collection cost
customer site	cycle time, rework cost, production defects	development independence, testability, reliability, scalability	development independence, testability, reliability, scalability	not easy		Very / A lot		your scalability metrics seem focused on performance. although they contribute to costs, they should not be a major consideration when migrating.	not easy	most of them. vast, heterogeneous ecosystem. Lack of baselines. Absence of numerical variable (e.g. testability)
Upper Manager	it was a new project/capability so it made sense.	memory	memory consumption			Very / A lot	mostly	not easy		reusability, complexity
Manager	before (extensibility, team alignment, technology independence), During (memory consumption)					Absolutely	yes	we think it is comparably easy to collect the technical factors (CPU, memory, response time ..). However, based on our experience it is a more difficult way to measure process and organisational related factors.		dependes on the size of the system. for smaller systems the measurement of all factors could be more costly than actual development. May be provide a different sets of metrics?
Developer	before (response time, system requirement), after (velocity, extensibility)					Very / A lot	yes	easy		
Manager	Before (complexity, #bugs), after (after (scaling, complexity, #bugs)					Very / A lot	yes	easy		
Developer	Before (LOC, change freq), after (scaling)			easy	none	Absolutely	yes	not easy		
Developer	development effort complexity (BOTH)					Absolutely		yes	easy	
Developer	respinse time, cost					Very / A lot	yes	easy		
Manager	infrastructure cost and development cost (both)					Absolutely	yes	very easy		
Developer	Before (defects), after (cost)	team_independence				Absolutely	yes	not so easy		
Developer	hard to scale(Before), velocity (Both), release time (after)	resource_utilization				Very / A lot	yes	easy		
Developer	downtime of the system, complexity, performance (Both)			no		Absolutely	yes	very easy		
Manager	extensibility (both), number feature blocked (Both)	deployment_time		no		Absolutely	yes	easy commonly	no	yes , why not
Developer	Complexity, bugs, development effort, velocity, Loc (Both)					Absolutely	yes	easy	no	yes
Developer	coupling, scaling, release time	scalability				Very / A lot	yes	easy		
Developer	#bugs, development, #effort defects (both)	team_independence				Absolutely	yes	very easy		yes
Developer	Before (development effort, system requirements, technology independence), after (bugs)					Very / A lot	yes	easy		yes
Developer	time of scaling, easy of scaling (both)	team_independence				Very / A lot	yes	easy		yes
Developer	extensibility, velocity of the system, release time of the new features (both)	cost, deployability, maintainability		metrics related to automation	automation related metrics	Absolutely	yes	very easy		yes
Manager				no		Very / A lot	yes	not so easy	metrics related to automation	
Developer	#bugs, #LOC, complexity, time to release (both)					Absolutely	yes	easy	no	yes
Developer	response time, #bugs, memory consumption (after and during)					Absolutely	yes	easy		yes
Developer	complexity , #bugs (Before and during)	deployment_time				Absolutely	yes	very easy		yes
Developer	response time, memory consumption, development effort (before and during)					Very / A lot	yes	easy		yes
Developer	change frequency, ,maintainability, development effort, #defect, #bugs (before and during)					Absolutely	yes	very easy		yes
Developer	maintenence effort and complexity (before and during)			no		Very / A lot	yes	easy		yes
Developer	delivery time change frequency					Very / A lot	yes	easy	no	yes
Developer	Development effort, #bugs (before and during) scaling (during)					Very / A lot	yes	easy		yes
Developer	complexity, #defects, # bugs, changeability					Absolutely	yes	easy		yes
Manager						Absolutely	yes	easy		yes
Developer	development effort, maintenance effort (before and during)#bugs (during)					Very / A lot	yes	easy		yes
Developer	Maintenance effort, #bugs, delivery time, complexity, #defect (both)					Absolutely	yes	easy		yes
Manager	complexity, #bugs (both)		employee morale			Very / A lot	yes	easy		yes
Developer	Page speed and user downtime. We were having very high response times and many severe outages.	scalability				Absolutely	yes	We use some tools that help us gather these metrics, such as NewRelic. So, not too bad.		yes
Architect	Number of dependencies between teams, impact of failures, number of deployments,	maintenability	costs			Absolutely				
Upper Manager	Profitability of the system					Very / A lot	Yes.	Need a working prototype to measure		Probably