TaP Software Factory Gaps to Insights Insights to Desired Experience

**Experience to Business Process** 

Differentiation, Path-to-Profit, Scaling & Competence

Requirements to Capability, Feature, Epic and Stories

User Experience & Interface Design Specifications & Requirements

Business Dimension Capabilities Dimension Customer Dimension Stories to coding, bug-fixes, testing and deployment (T8) Business Process & Data (T9) Capability, Feature, Epic, (T3) Ideate & Discover (T7) Feature Requirements (T12) Check (T13) Action (T1) Empathise (T2) Define (T4) High-Level Prototyope (T5) Test & Iterate (T10) Plan (T11) Do <u>Inputs</u> Inputs & Outputs Ouput of (T9) Output of (T10) Output of (T11) Output of (T12) Observations Gaps: Behaviours, Insights & problem Potential solutions & Prototype for Demo Desirable solutions & Customer Journey Map & Feature Requirements, Detailed step-by-step Level Data Input and Output draft UI design, inputs, <u>Outputs</u> **Outputs** <u>Outputs</u> <u>Outputs</u> pain-points, unmet needs, statements customer experience new experience 2 business process Qualitative Research **Outputs** habits, expectations <u>Outputs</u> outputs, logic Technical requirements sprint plan, sprint goal lessons learnt, defect Assess change impact <u>Outputs</u> <u>Outputs</u> <u>Outputs</u> <u>Outputs</u> Feedback & Refinement, <u>Outputs</u> new? (i.e. APIs, grooming stories, code, analysis, root cause, processes to modify, **Outputs** Feature Requirements, **Outputs** Potential solutions & Wireframe, Clikable Customer Journey Map & Detailed Epic & Story as Desirability and Viability of Gaps: Behaviours, Architecture, Db), sprint review, demo, recommended changes, training impact, deploy Insights & problem draft UI design, inputs, customer experience Prototype, Simple Journey, detailed Data Input and Detailed step-by-step Level input for software pain-points, unmet needs, solutions and experience dependency analysis, feedback and input, update update program stats, code, production release statements outputs, logic, T-shirt sizing T-shirt sizing (1st) Output Requirements 2 process, final UI design, development habits, expectations security, sizing estimate, backlog health & KPIs, Impact to checks & approval (2nd), one-liner stories better expressed stories <u>priority, release plan</u> timeline & cost Gain valuable input **Convert the requirements** Ideate new ideas to solve Plan the development Plan and execute the Determine the gap and Derive new insights and Evolve the solutions into Test and Iterate the Map the entire customer Document the required Map the entire business Deploy the changes by define the root problems. prototypes that can be into Epics and Stories, to sprints to deliver the about what the customer the root problems. prototypes, improving as Journey in detail to features, including the requirements from the the root causes of the modifying process, process to support the tested with the customer. detailed data fields and modules required. training. Deploy code enable the detailed new experience desired. enable development to gap from output to epics needs. you progress Epics and stories. 1. Discuss ways to solve the 1. No new insights = same L. Translate solution or & stories to feature requirements to logic required. 1. Develop the sprint plan, into production. 1. Improve the fidelity of the 1. Target segment experience as before. root problem(s). 1. Document the business 1. Develop the technical experience (many solutions) Purpose be mapped. requirements. identification 2. Observations are *what* 2. Diverge sufficiently prototype as you iterate. 1. Document the required process with a process 1. Detail the requirements, requirements. determine the sprint goal 1. Make the necessary into a lo-fi or mid-fi prototype. 2. Record the customer's 1. Walk though and 2. Detail the business logic 2. Code and demo the 2. Observations of you see. Insights are the before converging. features to enable the flow diagram with and organise by epics and 1. Document the gaps changes to process and 2. Create the end-to-end document the end-to-end stories. module. between the desired output | training. customers, in their reasons why they happen. 3. If you are stuck, find experience with the customer journey. swim-lanes and owners. required customer journey 2. Refine the draft UI into 2. Detail the business logic 3. Detail the data inputs. 3. Obtain feedback and and the current output. environment, encountering 3. The root problem ways to introduce greater prototype, solicit the customer journey. 2. Draft the UI design. 2. Secure production 3. Estimate approx man-days 2. Determine the root cause | release approval. input from key yet unseen problems to statement is the root variance into the input. provide detailed data fields the final UI after the required outputs and intermediate feedback and refine. 2. The journey will capture to develop the module to reveal hidden needs reason why they happen. 4. Connecting different 3. Check the desirability of everything the customer for input and output, and business process mapping 3. Detail the data inputs, data required. stakeholders. and attribution. 3. Release code into produce the prototype outputs and intermediate 3. Place your self in the 4. Solutions are how to the solution or experience will experience from start to the required business logic. is completed 4. Update the backlog 3. Analyse and report the production solutions to problems in the experience. shoes of the customer to same "job" creates an by asking willingness to finish. 3. Map all the new and data required. based on the gap. impact. solve the problem, which is understand problems not addressed now. pay and the NPS question existing data required 4. Take remedial action to experience. improve the process. encountered 1. The focus in this stage is to 1. There must be sufficient L. Check for clarity of the 1. Test to understand for 1. The end-to-end customer 1. Ensure **all relevant units** . **Document** the required 1. Sprints are time-boxed 1. This last stage **evaluates the** 1. Wallet Size of target L. Insights are the reasons 1. What changes and . The technical requirements understand the **deviations** and periods of one week to one customer instructions by Epics and Stories using the **changes** required and the problems with root causes **usability**, i.e. is the customer journey is the **totality of the** are represented and a why customers behave the dependencies are required in are **only as good as** the segment relevant to your month, during which a product impact of the change. reviewing them with that **haven't been solved** to able to navigate the journey **experience** the customer cross-functional team is artifacts. way they do. They need to other capabilities that stories, business logic and products or services. 2. The root cause is analysed owner, scrum master, and 2. If the changes are approved start on ideas and solutions. someone that has not been intuitively without instructions. enjoys. involved in the business 2. Ensure that all information 2. Total no of customers in meet the next 4 criteria. input and outputs provided. already exist? scrum team work to complete a and **changes** may be processes are modified, involved. 2. Eye-tracking software can 2. If insufficient or unclear 2. Customer interactions will process and data mapping. that is mandatory is filled in. the target segment. 2. Explains an observation 2. The UI design can begin 2. Determination of what **new** specific product addition. recommended to the PDCA training **updated and** 2. Decide what needs to be inputs, gather more gaps and be used to determine the consist of multiple customer 2. Start by agreeing on a 3. It's a good practice for the 3. Wallet size x no of about a behavior that most once the customer journey is core system capabilities are 2. Grooming involves splitting process or at stages (T1) to introduced to the affected tested to **ensure the concept** Considerations insights rather than rush. **clarity** of the on-screen journeys, e.g. onboarding, maximum elapsed time for party documenting to **play** customers = Revenue (must people may not have heard required, and hence **new** firm. Good best practice to big stories into smaller ones, (T9) if thats where the issue **is soun**d, vital for anything 3. Insights are more likely to instructions. selection, shipping details, the process end-to-end. back the requirements to be sufficient) before, i.e., not have UI specialists from the APIs, changes to architecture rewriting backlog items to be 3. The changes are an materalize if you combine that is new or changed. 3. Test to ensure that the payment, tracking, etc. 3. Use **Lean methods** to the product owner to ensure 4. Minimisation of bias in commonsense or expected. databases, etc., is a key development team comment more expressive, deleting **3. Attribution** back to the important part of the 3. Start with a low fidelity data and observations **from** offering solves the 3. Map the **data and business** remove waste in the process: correct interpretation on feasibility of the Ui design. selection or answers. 3. The observation must be requirement of this stage. obsolete items no longer stream, unit or department is continuous improvemnet **logic** needed to complete the 3. How do you ensure that all prototype to **get the basic** different vantage points customer's problem waiting times, defects, motion 4. Doing this now, will **prevent** 5. No of unresolved connected to a customer 3. An analysis of needed, etc. key to ensure learnings are process. concepts right. rather than just one. entirely. journey at a high-level. (e.g. hand-offs), etc costly reworks later. problems identified. problem or delighter for business logic is captured dependencies on other embedded in the source of the 3. A sprint plan and goal are 4. The code is **deployed** after 4. Determine the **high level** 4. If there are too many ideas 4. Understand the customer's 4. Ask the customer for 5. Lack of specific details in 6. Importance of these which they are likely to be development, whether other correctly, as this is an area created. deviation or defect, without all fixes are done, and the afte or good solutions, prioritize data mapping, i.e. what data willingness to pay for such information that **aligns with** the story affects the entire that often introduces change **problems**, so that there is willing to pay for. core systems, other epics, focusing on blame. 4. Development work begins the required production inputs are needed from the them according to **impact** an offering. the way he retrieves the development, as is **one of the** willingness to pay for their 4. Offers a simple, clear and stories, etc., is key to 4. The program key **statistics** thereafter (note that this is release permissions are elegant explanation of why a  $\mid$  and cost in (T4). customer, what are needed 5. Understand the **profit** information **leading causes** of delay and solutions. 4. Obtain the **meta-data for all** understanding the effort are updated, e.g. cost, ohtained actually a **small part** of a customer behaves the way he |5. Before entering into (T3) from the business, and dynamics of the entire 5. Handle straight-through cost over-runs. **7. Gaps** represents the data required from the required. timelines, bug fixes, etc. huge preparation, planning, gather all the information you document the **high-level** offering (product, service & green flows as well as difference between the database. Ensure that the 4. Once the above have been does. testing, deployment effort) business logic required. can about **how your** experience) to ensure that the non-green flows with completed, sizing of the current and the ideal. 5. You can **act on and exploit** length of fields match those i 5. Demonstrations, feedback **competitors** are solving the offering is viable, and exception handling. development man-days can the insight to derive a 8. Behaviours that become the database to prevent esign and stakepholder inputs are eventually you are able to turn 6. Develop KPIs and a system problem (and if they are not). automatic are habits. **proposition** that can be done. truncation. obtained, and attendence is to **measure and monitor** for a profit. differentiate your offering. 5. The effort is then compared key so as not to discover the defects. 6. Problem statements must to other priorities and deviations later, when they 7. Analyse root cause of be the **root problem**, and not **trade-offs made** to sequence are expensive to correct. process defects and and schedule the feature, epic a symptom. eliminate them. or story for development. Iterate between (T6) Customer Journey Mapping, (T7) Feature Requirements, (T8) Business Iterate between (T1) Define and (T2) Empathise Iterate between (T3) Ideate & Discover, (T4) High-level Prortotype & (T5) Process & Data Mapping & (T9) Documentation of the Capability, Feature, Epic and Stories as Agile software development methodology until you have sufficient insights and a clear, Test and Iterate to ensure prototype is desirable (i.e. highly usable & input (T10) - (T13) to ensure all business logic & data requirements are clearly identified & solves the root problems) & viable (i.e. able to make a profit sometime) root problem statement. documented. minimum requirements checklist Input & OUtput roduct & Service Development Responsibilities (Insert Journey & Data & Process Mapping Roles Agile Software Development Team