Memo 02 - How to Work Well on Teams and Build the Cross-Functional, Collaborative Team

While reading chapter two of both Software Engineering at Google: Lessons Learned from Programming over Time's "How to Work Well on Teams" and Create Your Successful Agile Project's "Build the Cross-Functional, Collaborative Team", respectively, provides guidance and insight to how work is done from an analytic perspective (how a team collaborates and can be cross-functional) and holistic/relational perspective (how one fits/conforms within a team).

A functional team incorporates both perspectives and efforts to improve not only as an individual but also as a collective. The drive to push oneself to embrace collaboration, and the need to cultivate and encourage teamwork within the team/company operates from both bottom-up (self-development and training) and top-down (company practices and policy) approaches.

In the Software Engineering at Google: Lessons Learned from Programming over Time's "How to Work Well on Teams" one's insecurity fearing putting oneself at risk of failure and/or cheating one's potential for growth within the team/company; as a result, they hide what they know, silo information, from the rest of team or external groups. Top down in and Create Your Successful Agile Project's "Build the Cross-Functional, Collaborative Team" this is a trap as the Team member(s) has a history and culture of individual work, not collective work.

The one harm in a non-collaborative, fear to share with others, work environment "How to Work Well on Teams" calls to attention is "The Bus Factor" where in a hypothetical scenario an essential/key value employee is hit by a bus all the knowledge and skill is lost in the team for the duration of recovery. Siloed knowledge hindered the team from being productive and operate as efficient as it should be. This siloed knowledge isn't just purely technical, but also include simple everyday things like passwords, location of keys, the time of meeting with customers.

I can resonate with this pain for when I transferred from a previous role into a new position the team member who was to be my mentor and training buddy, after only three days of training, left for his sabbatical. Although the team knew that he was going to be on leave no one else had known the password to the onboarding file-share, nor did they know where the key was kept to the lab's testbench systems.

This could have been fixed by having an internal wiki-page where best-known-methods (BKMs) and key information can be easily accessed.