**🧾 Résumé – Jie Zhang**

**📌 Personal Information**

* **Name:** Jie Zhang
* **Gender:** Male
* **Date of Birth:** October 1988
* **Place of Birth:** Shanghai, China
* **Email:** [sagibrant@163.com](mailto:sagibrant@163.com)
* **Phone:** +86 136 1167 2073
* **LinkedIn:** [linkedin.com/in/sagi-zhang-2845451b](https://www.linkedin.com/in/sagi-zhang-2845451b/)

**🎓 Education**

| **Period** | **Institution** | **Degree & Major** |
| --- | --- | --- |
| Sep 2011 – Jun 2014 | East China Normal University | M.Sc. in Computer Software and Theory |
| Sep 2007 – Jun 2011 | East China Normal University | B.Sc. in Software Engineering |

**💼 Work Experience**

| **Period** | **Company** | **Role** |
| --- | --- | --- |
| Jan 2023 – Jun 2025 | OpenText | Manager I |
| Jul 2017 – Jan 2023 | Micro Focus | Manager I |
| Nov 2015 – Jul 2017 | Hewlett Packard Enterprise (HPE) | Senior Software Engineer |
| Jul 2014 – Nov 2015 | Hewlett Packard (HP) | Software Engineer |

**🧠 Technical Expertise**

* **Languages:** C++, C#, JavaScript
* **Technologies:** Desktop App, Browser Extensions, Web Automation, etc
* **Management Skills:** Six Thinking Caps, SLII Leadership, HBDI, etc

**🌟 Recent Highlights**

* **Hybrid OCR Solution:**  
  Engineered a hybrid text-recognition system that drastically improved local OCR accuracy for AI automation.
* **Coordinate Transformation Algorithm:**  
  Built a cross-scale algorithm to map screen coordinates to viewport coordinates—solving persistent UI inspection offset issues.

**🧾 Patents**

* [**Automated Identification of Input Elements in a GUI**](https://patents.google.com/patent/US20220107883A1/en)  
  *Inventors:* Peng-Ji Yin, Cheng Hua, Jie Zhang
* [**Recommending Programmatic Descriptions for Test Objects**](https://patents.google.com/patent/US11698851B2/en)  
  *Inventors:* Peng-Ji Yin, Cheng Hua, Jie Zhang

**🚀 Professional Summary**

Accomplished Software Engineer and Team Manager with over 10 years of experience leading the evolution, modernization, and optimization of the UFT/QTP platform. Recognized for delivering scalable, complex solutions, resolving critical CPE incidents, spearheading the development of core features, driving innovation in automation tooling, and fostering collaboration across globally distributed teams.

**💡 Core Strengths**

* 🌐🧠 **Cross-Team Collaboration:**  
  Successfully integrated with ALM, MC, SRF, and AI teams across continents to deliver seamless functionality.
* ☁️⚙️ **Cloud & DevOps:**  
  Rapidly adopted and applied TypeScript, AWS, and Kubernetes technologies in SaaS test execution platforms (SRF).
* 🧹⚡ **Codebase Refactoring:**  
  Tripled performance in UFT Web by merging multi-request logic into efficient "atom" messages after analyzing millions of lines of legacy code.
* 🤖🔍 **AI Feature Development:**  
  Solved long-standing OCR accuracy issues in on-prem deployments by pioneering a hybrid text identification strategy—outperforming legacy engines (ABBYY, Tesseract) where conventional AI solutions had failed.
* 📐🖱 **Algorithmic Design:**  
  Developed a precise screen-to-viewport coordinate conversion mechanism to overcome long-standing test automation limitations.

**🛠 Projects**

**🧠📦 Unified Functional Testing (UFT/QTP) - *2014.7-2025.6***

As a **Core Developer** and **Team Manager** for **Unified Functional Testing (UFT/QTP)**, I lead the team in delivering new features, handling customer issues (CPE incidents), and driving architectural modernization and innovation.

**UFT** (formerly **QTP**) is a flagship commercial test automation platform:

* 🏛️ **Legacy & Evolution:**  
  Born as *Astra QuickTest* in 1998 (Mercury), the product evolved through the HP/HPE, Micro Focus and OpenText eras into today's AI-powered UFT One/OpenText Functional Testing—delivering script-based and AI-driven automation and ui-testing capabilities.
* 🧩 **Codebase Scale:**  
  UFT is built on a massive architecture comprising hundreds of millions of lines of code across C++, C#, and VBScript, with deep integrations into Windows APIs and COM-based technologies.
* 🌍 **Enterprise Adoption:**  
  Trusted by more than 2,800 global enterprises, particularly in Fortune 500 verticals such as finance, healthcare, and government sectors.
* 📊 **Market Share:**  
  Holds approximately 2.14% of the $22.2B automation testing market, generating hundreds of millions USD in annual revenue.

**🔹🤖🧠 UFT OCR Improvement – *2024-2025***

Legacy OCR engines such as ABBYY and Tesseract frequently underperformed in on-premises environments, falling short of the accuracy needed for reliable AI automation. Additionally, many customers were unable to use cloud-based OCR services due to privacy constraints or infrastructure limitations.

To address these challenges, our team evaluated five potential solutions—including EasyOCR and enhanced Tesseract variants. Among them, the initial proof of concept (PoC) for a Hybrid Text Identification approach was initially rejected due to slow performance (~20 seconds per scan) and lack of word segmentation.

After I joined the initiative, I optimized the algorithm to reduce the processing time from **~20s** to **~2s**, while also enhancing the extraction logic to include word boundary segmentation. As a result, my PoC was selected and successfully implemented.

The Hybrid Text Identification solution intelligently fuses OCR results with DOM-based web text extraction. Key challenges included handling complex styles, nested iframe structures, and ensuring minimal performance impact during test execution. This hybrid method significantly outperformed traditional engines, achieving up to **100%** accuracy on web-rendered text—substantially improving automation reliability in secure, offline environments.

**🔹🌐🧠 UFT Cloud Browser Architecture Redesign – *2024***

The original UFT Cloud Browser architecture, built around a lightweight Java connector and direct WebSocket communication with browser extensions, lacked the flexibility and integration depth required by modern AI automation use cases.

A key limitation was its incompatibility with UFT AI’s native Windows-based architecture, which relies heavily on COM objects and Windows message communication. To bridge this gap, I led a comprehensive architectural redesign of the Cloud Browser platform.

Key contributions include:

* **Redesigned the communication framework** to support a modular, protocol-agnostic message system compatible with standard I/O, Native Messaging, and web requests—while maintaining alignment with UFT's existing messaging conventions.
* **Rewrote the input simulation modules** from the ground up to accommodate AI-driven interaction logic.
* **Built a new execution engine** that orchestrates complex automation steps and delegates execution across isolated driver processes—enhancing maintainability and parallelization.

The result is a robust, scalable Cloud Browser platform that seamlessly supports UFT AI automation workflows and meets enterprise-level stability and extensibility requirements.

**🔹☁️🧪 UFT Cloud Browser Quick-Win Initiative – *2023***

Following the sunset of the StormRunner Functional (SRF) platform, market demand for a cloud-based browser automation solution remained strong. However, due to tight budget and resource constraints, stakeholders sought a lightweight alternative with minimal development overhead.

As the technical lead, I architected a lean Cloud Browser solution in collaboration with the Digital Lab team, achieving rapid delivery with strategic reuse and design simplification:

* **Leveraged** existing Digital Lab infrastructure and UFT browser extensions to minimize new implementation overhead.
* **Redesigned** the communication layer using WebSocket-based messaging, replacing complex legacy pipelines with a streamlined, maintainable protocol.
* **Reimplemented** key automation features using browser extension APIs, eliminating dependencies on Windows-native APIs for improved cross-environment support.
* **Established agile co-development practices**, including weekly syncs and daily hands-on code reviews with the partner team.

This effort resulted in a fully functional UFT Cloud Browser released in just **4** months with only **4** developers, dramatically outperforming the prior SRF initiative that required **3** years and **~30** engineers—demonstrating our engineering agility, architectural foresight, and ability to deliver high-value solutions under tight constraints.

**🔹🧩🧪 UFT MV3 Extension Migration – *2022–2023***

With the deprecation of Manifest V2 (MV2) by Chromium, I led the successful migration of UFT's browser extensions to Manifest V3 (MV3) to ensure continued compatibility with Chrome and Edge.

Key technical contributions included:

* **Redesigned the extension messaging architecture**, replacing legacy window.postMessage patterns with modern Extension APIs for enhanced security and maintainability.
* **Implemented an asynchronous frame hierarchy construction mechanism**, ensuring non-blocking JavaScript execution within content scripts—crucial for high-performance automation scenarios.
* **Engineered a custom user script evaluation pipeline** to bypass MV3’s stricter Content Security Policy (CSP) restrictions, allowing automation to continue handling href="javascript:" link interactions where default click simulation fails.
* Ensured **full backward compatibility** while upgrading over **100,000+** customer environments, with no disruptions and noticeable performance gains.

This work safeguarded long-term extensibility and performance of UFT's web automation capabilities under the evolving Chromium extension ecosystem.

**🔹🧩⚡ UFT Web Fast Run Mode – *2020–2021***

As modern browsers evolved and customers transitioned away from IE, the legacy UFT Web Add-in architecture—originally optimized for IE—began to show severe performance degradation, particularly on complex web pages. This was exacerbated by the shift from native C++ extensions to JavaScript-based extensions, as well as the migration of communication from direct object access to a bridged messaging model.

One major enterprise customer reported a critical slowdown: test steps using Virtual Relation Identification (VRI) degraded from 3 seconds to over 3 minutes, severely impacting usability.

To address this:

* 🔍 Conducted deep analysis across **400,000+** lines of legacy code to trace performance bottlenecks and architectural inefficiencies.
* 🧠 Identified key issues: excessive attribute query messages, redundant existence checks, and high I/O overhead due to fragmented communication.
* 🛠️ Designed a new UFT Web protocol layer, introducing a unified atom message format that encapsulates all step data in a single request—replacing 10–60 fragmented messages per step with a single round-trip.
* 🚀 Resulted in **3×** average performance improvement on standard browsers and **5×** improvement in WebDriver-based environments (e.g., test runtime reduced from 20 minutes to 4 minutes).
* 💡 Specifically, for the VRI-heavy scenario, execution time improved from 3 minutes to 5 seconds.
* 📘 Beyond technical gains, this initiative helped upskill the team and deepen collective understanding of the 15+ year-old UFT Web codebase.

This deep refactor not only boosted performance but also enhanced team understanding of the 15+ year-old codebase, enabling more confident modernization in future cycles.

**🔹🧠🔤 UFT TextObject Feature – *2019***

Traditional UFT TestObjects are tightly coupled with specific automation technologies such as Web, UI Automation, Java, .NET, and WPF. However, real-world scenarios often expose environments where these technologies fail due to limitations in hooking or restricted access to application internals.

UFT previously introduced the InsightObject, which used image comparison to simulate interactions. While useful, its reliability degraded under variable screen resolutions, DPI settings, or scaling factors, making it unsuitable for many customer environments.

To address this, I proposed and designed a new kind of TestObject—TextObject:

* Utilizes OCR-based text recognition to locate UI elements by visual text instead of DOM or accessibility hooks.
* Simulates user interactions (keyboard/mouse) based on the text's screen coordinates, effectively bypassing traditional element hierarchies.
* Offers a robust fallback where injection-based or native API automation is not feasible, especially in secured environments that prohibit hooking technologies.

This innovation enabled large, security-conscious organizations to continue functional testing and automation without compromising infrastructure policies—bridging a critical gap in UFT's automation capabilities.

**🔹🧪⚙️ UFT Parallel Execution – *2018***

Historically, UFT (Unified Functional Testing) did not support parallel test execution—a major bottleneck for customers aiming to accelerate their automation pipelines. Despite UFT’s minimal memory (sub-200MB) and CPU footprint, each test instance required a separate machine due to architectural limitations.

This gap became critical when pre-sales teams reported losing deals to competitors who could run 10+ mobile tests in parallel on a single laptop.

**Technical Challenge:**

* UFT’s core consists of **4 million** lines of legacy code, heavily reliant on singleton COM objects and assumptions of a single instance per machine.
* Many modules were not designed to support concurrent execution contexts or isolation.

**My Contributions:**

* Leveraged prior experience from UFT Cloud Execution on SRF to develop a proof-of-concept for UFT Mobile Parallel Execution.
* Refactored legacy components, removing unnecessary dependencies while isolating the mobile execution logic into a standalone runner.
* Enabled 10+ mobile test instances to run concurrently in a single user session, maintaining speed, reliability, and resource efficiency.

**Outcome & Impact:**

* The POC received the recognition from the Direct manager and shifted product strategy: PM reprioritized the roadmap to focus on parallel execution.
* Led the team in launching the UFT ParallelRunner, later extended to UFT Web add-in and Java add-in.
* Empowered pre-sales to successfully compete in critical deals.
* QA teams experienced significant test cycle reduction, dramatically improving CI/CD throughput.
* Promoted to **Manager I** in recognition of this high-impact delivery.

**🔹☁️🧱 UFT Cloud Execution on SRF - *2016-2017***

StormRunner Functional (SRF) is a cloud‑based solution for executing functional tests on web and mobile applications. It provides a virtual lab with multiple browsers, operating systems, and mobile devices—allowing teams to run tests across diverse environments without managing infrastructure.

As a core developer on the UFT Backend team, I was responsible for designing and implementing the UFT Cloud Execution solution:

* **Initial AWS EC2 Implementation**  
  Delivered a working prototype that ran UFT One tests on AWS EC2 instances. While functionally complete, this approach incurred high costs and lacked scalability alongside SRF’s Kubernetes‑managed services.
* **Windows Container Migration**  
  Re‑architected the solution using Windows container technology to align with SRF’s K8s‑native infrastructure. Outcomes:
  + **75%** cost reduction (down to one‑quarter of EC2 expenses)
  + Fully horizontally scalable within SRF’s container cluster
* **Cross‑Region Collaboration**
  + Partnered with engineering and DevOps teams across multiple regions
  + Presented the proof‑of‑concept in Israel to senior pillar managers
  + Worked directly with the Israel DevOps team to integrate Windows containers into SRF’s Kubernetes pipelines
* **Business Impact**
  + Empowered UFT on‑premises customers to seamlessly transition to SRF’s cloud platform
  + Enhanced platform reliability and user satisfaction
* **Ancillary Revenue Stream**
  + Published the Windows container–based Docker image on Docker Hub, generating an unexpected $2-3 million/year in additional revenue

This initiative not only transformed UFT's cloud execution capabilities but also served as a blueprint for future containerized test automation solutions across the organization.

**🔹📱🔗 HP/HPE Mobile Center Integration - *2016***

HP/HPE Mobile Center (MC) is a mobile testing platform that enables QA teams to run tests on real iOS, Android, and Windows devices.  
I was responsible for maintaining the UFT–MC integration and developing new features to extend its capabilities.  
One of the key enhancements was adding MC Browser support, allowing UFT Web scripts to run on mobile browsers.  
During this effort, I collaborated with MC founder Ameer during his visit to Shanghai, where we discussed and resolved key design challenges.  
Based on the new design, we implemented a bridge service between two UFT add‑ins, enabling a single TestObject to function across both.  
This design laid the groundwork for enabling future support of Cross‑Browser Testing within UFT.

**🔹📄🧾 UFT HTML Report Logger - *2015***

Previously, UFT used the UFT Run Result Viewer (RRV), a legacy WinForms application, to display test run results. RRV was inefficient when handling large volumes of test steps and made sharing results difficult, as recipients needed to have UFT installed to view them. However, QA managers and stakeholders often needed access to these results without requiring the full UFT setup.

To address this, the FA and PM initiated the development of a new HTML Report, designed to be lightweight, shareable, and easy to view in any browser as a single self-contained .html file.

I collaborated with another team member to implement this solution. I was responsible for developing the HTML Report Logger, which listens to events from the UFT ScriptEngine, collects the necessary test step information, and writes the data into a structured JSON file. My teammate then handled the HTML generation in a separate process—styling the UI with embedded CSS and rendering the report using the JSON data.

Through this project, I deepened my understanding of the UFT Execution Engine and became a key backend expert within the team.

**🔹🖥️🔌UFT Remote Agent - *2014***

UFT Remote Agent is a DCOM component used to enable remote test execution from ALM/QC (formerly Quality Center). Originally developed in C++ many years ago, the component had evolved into a monolithic class of over 9000 lines of code. This made the system difficult to maintain, with frequent customer-reported bugs and limited modularity.

Due to these challenges, the Software Architect decided to refactor the component using C#. I was responsible for leading this effort.

I performed an in-depth analysis of the original codebase and worked closely with the ALM team to fully understand the integration points between UFT and ALM. Based on this understanding, I redesigned the class structure by separating responsibilities according to UFT’s functional areas. I also rebuilt the Remote Agent UI using WPF for a more modern and maintainable interface.

One of the key challenges was performance: the new C# implementation initially lagged behind the original C++ version. I spent about a month optimizing performance using Visual Studio Profiler and implemented various improvements, including loading the UFT AOM Agent as an assembly to reduce cross-process communication overhead.

In the end, the new UFT Remote Agent achieved comparable performance to the legacy version, delivered a cleaner UI, and significantly improved maintainability. For this work, I received my first **Team Star** award.

**🌍 Other Notable Projects**

**🔹🇧🇮📊 Burundi Population Census System • *Jul 2013***

Contributed as an independent developer to the design and deployment of a national data collection system, supporting Burundi's presidential campaign. Helped improve system architecture and deployment strategy for field use.

**🔹💧📡 Troll – Groundwater Monitoring • *May 2011***

Joined the team as an independent contributor. Developed a long-term environmental data monitoring solution featuring robust 3G data transmission. The system operated reliably with minimal maintenance for over 4 years.

**🔹🍽️💳 Botion Restaurant System • *Aug 2010***

Joined a startup (xpfood.com) and led the end-to-end development of a CRM and POS system, including invoicing and card-based customer features. Delivered a complete, production-ready solution for restaurant management.