automation hero_

The Technology Behind Automation Hero

To automate business processes with AI, business workflows need to be driven through AI models that were trained on historical data and that are continuously improved and refreshed. Traditional business process automation platforms are modeled manually and are less powerful and efficient than AI models that are trained on historical data. Individual steps such as updating a customer address or creating a quote are done manually by the human workforce.

A modern AI automation platform should drive and automate these steps through AI. The human workforce will remain a critical component in the workflow, but the real goal is to automate as many steps with AI – from data entry to reaching out to customers at the perfect time – as possible. At this point there is no platform in the market that combines a fast data platform with AI to drive business workflows.

Automation Hero was developed from the beginning as a new business operating system combining fast data, AI and business process management capabilities to form an intelligent business automation platform optimized for workloads in a modern and AI-focused world. We call this platform Hero_Flow.

Reactive Platform

Hero_Flow is implemented from the ground up as a Reactive System. There were four major goals for this architecture:

1. **Create a system that responds quickly and consistently to client requests.**
2. **Build a resilient platform with a sustainable infrastructure, hardware, network, user and code errors to maintain responsiveness.**
3. **Make an elastic system that maintains responsiveness even as workloads vary.**

Maintain an asynchronous (out-of-sync) message-driven approach to allow loose coupling and location transparency for all components to distribute data to a flexible group of processors.

Gossip Protocol

Hero_Flow is built with a gossip communication protocol, which means that rather than one central entity communicating with thousands of nodes, each node communicates with other nodes at random to quickly spread information at scale. Thus Hero_Flow is resilient and can overcome network issues and node failures.

Eventual Consistency / Conflict-Free Replicated Data Types

With Hero_Flow's decentralized gossip protocol, eventually, all nodes receive a data update. Because there's no maximum number of nodes the data can be read from, as the number of nodes increases the likelihood of getting outdated data decreases.

Hero_Flow uses a scalable and performant approach of conflict-free replicated data types for distributed state management. The goal is that a data type that also contains its history can be merged without conflict.

No Single Point of Failure or Centralized State Management

Using gossip protocol, eventual consistency and conflict-free replicated data types, Hero_Flow avoids failure from a single point and implements a fully decentralized state management. In case of a cluster partition or hardware failure, the split-off cluster segment can operate autonomously and later merge back into the cluster without conflict. Hero_Flow operates with large clusters distributed over many racks and data centers to achieve high availability with built-in disaster recovery mechanisms.

Scalable Actor System

To achieve a reactive (responsive, resilient, elastic and message-driven) system, Hero_Flow is built as an Actor System in which each component is an actor that communicates with other actors through asynchronous messages.

Distributed – Java 9 streaming

Java 9 Reactive Streams formalized a universal standard to process data through a set of processors connected by a publish-subscribe mechanism, which perfectly matched our design goals. The advantage of this is the non-blocking implementation of back pressure and that the platform can pin certain data flows or processes for hardware groups. For example, it's possible to pin a data flow to a set of nodes with GPUs to accelerate Google Tensorflow execution.

Dynamic Partitioning / Elastic Auto Scaling

Hero_Flow is built from the ground up to work in an elastic environment with a continuously changing number of nodes, so the system can dynamically partition data to parallelize processing in a number of different ways. It's possible to choose from different strategies like partition data based on currently available cluster resources, defined partitions, increased cluster resources, or dynamically expand and contract using a work-pulling pattern.

Performance and Utilization

The runtime on every node of Hero_Flow is minimal and optimized to use as little CPU, memory and IO as possible. On a single core of a 2.9 GHz Intel Core i7 CPU, Hero_Flow achieves 1.9 million 100 byte messages per second without any IO or CPU boundary. Hero_Flow uses the fastest in-class object serializer for complex objects by more than 40% without the need to manually define data types and generate Java classes.

To optimize hardware utilization, Hero_Flow can pick nodes in the cluster based on equal CPU or memory selection profiles and deploy data flow partitions on nodes that had lower-than-average CPU utilization over the last few minutes. To react to business events, Hero_Flow spans new data flows within a few hundred milliseconds. To ensure responsiveness, Hero_Flow expands the cluster size by interacting with AWS scale groups, Mesosphere, Hadoop Yarn or Kubernetics, predictively or reactively (for example, should the average CPU load exceed defined thresholds).

Security

Automation Hero understands the complex security needs of a data platform to fulfill internal and regulatory requirements. While some implementations need to be optimized for security and some for performance, Hero_Flow allows granular configurations of individual security capabilities, including:

- Pluggable encryption algorithms and strength
- Encryption of data from source to data flow
- Encryption of data from data flow to sink
- Encryption of any temporary written data
- Encryption of all communication between the node if configured
- Https access for the web-based admin console
- Integration into authentication providers such as OAuth, OAuth2, LDAP or Microsoft Active Directory
- Audit and access logs
- Encryption key rotation (on request)

Advanced Metrics & Monitoring

Hero_Flow provides advanced metrics and flexible monitoring integration with a lightweight footprint on the platform. It provides node health and node utilization metrics, in addition to throughput, back pressure and health metrics for every function, source and sink currently running in the environment. This allows instant pinpointing of cluster hardware, network, performance or custom code issues. A broad set of monitoring system connectors to collect, visualize and create alerts is available. Connectors include:

- JMX
- UDP based StatsD
- HTTP(S) REST API
- Graphite
- Datadog and others (on request)

Connectors

Hero_Flow is enabled with an easy-to-extend connector framework to integrate with over 50+ connectors for any kind of data storage to read or write data, including:

- Salesforce
- Gmail
- GCalendar
- IMAP
- Office 365 / Outlook
- SAP NetWeaver
- MongoDB
- Elasticsearch
- Casandra
- HBase
- Apache Kafka
- AWS Dynamo DB
- Hadoop Distribute File System
- AWS S3
- AWS SQS
- Google Cloud Pub/Sub
- Azure Storage Queue
- FTP
- HTTP
- JMS
- UDP
- MQTT
- Mainframe (on request)
- JDBC (Oracle, MS SQL Server, Teradata, DB2, MySQL, Postgress, AWS Redshift, SnowFlake etc)

AI Capability

Hero_Flow tightly integrates with Google Tensorflow to train and execute a wide range of deep learning models. Existing models can be installed and used as a function in any data or business process workflow and take advantage of the enterprise readiness of the Hero_Flow platform. Hero_Flow can train pre-defined (e.g. in Python) models or Hero_Flow's AI design studio can create deep learning models. The platform comes with three advanced deep learning-based engines:

1 Recommendation Engine

Hero_Flow's recommendation engine is used for cross and upsell, churn prediction and best next step suggestions. Our engine shows best-in-class performance using the latest innovations in the deep neuronal network space. It provides data flexibility to create user and item features. A customizable representation function can convert any kind of data, including behavior, time series and low-dimensional vectors. The recommendation model is then trained with a customizable loss function depending on the problem space.

2 Intent-detection Engine

The Hero_Flow intent detection engine can identify the intent of written communication including emails, text messages, or online forms and posts. Intent could be scheduling a meeting, returning a product or changing an address. Using novel concepts from transfer learning, the Hero_Flow intent detection engine advantage reaches high accuracy with small dataset training. The underlying sequence-to-sequence model is language independent and can be trained for most languages.

3 Dark Data Extraction Engine

Hero_Flow's dark data extraction engine can be trained to extract structured data from unstructured data. It can extract names, job titles, phone numbers, address or product data from text conversations, it also can be extended to first convert (e.g. image data to text) and then extract meaningful structured data. The language-independent model can train a wide set of named entities and use a performance-optimized sequence-to-sequence deep neuronal network.

Bringing Humans and AI Together for Maximum Productivity

Hero_Flow allows at any given stage of a data or business workflow to integrate human validation or motivation processes. It's personified user experience is an AI assistant and uses a gamified, engaging approach to motivate the user.

1 Intelligence Fabric

As data flows through data preparation, analytics and AI micro-service functions, Hero_Flow connects data sources, business systems, business processes and people to form an intelligence fabric that automates business processes and augments decision making. As Hero_Flow is deployed, this fabric of AI business automation can be iterated on. By continuously adding more data sources and leveraging feedback loops, the precision of the AI models will significantly improve.

2 Deployment Cloud / On-premise

Platform nodes are simple Java processes and can run on top of popular cluster resource management frameworks such as Hadoop Yarn or Mesos. The platform can also be native or containerized and be run and managed as an AWS scale group or Docker container in systems, such as Kubernetes.

3 Customization

A set of Java APIs is available to implement custom inputs, outputs and functions. Building these functions is simple as we provide single JVM runtime to test customizations. Developer training and consulting are available on request to ensure teams are up and running quickly.