

IBM watsonx Orchestrate - External Data Integration

Overview

Chain AI now automatically injects real-time crisis data from ReliefWeb and OpenWeatherMap into IBM watsonx Orchestrate agents as **context variables**. This allows your agents to use live humanitarian crisis data and weather conditions when generating responses.

How It Works

1. Data Sources

- **ReliefWeb API:** Humanitarian crisis reports, disasters, and supply chain disruptions
- **OpenWeatherMap API:** Real-time weather data with logistics impact assessment

2. Automatic Context Injection

When a user sends a message to the IBM watsonx agent:

1. The `pre:send` event intercepts the message
2. Fetches relevant data from both APIs based on user query
3. Injects the data into watsonx context variables
4. Agent receives enriched context alongside user message
5. UI panel displays the same data for human visibility

3. Context Variable Structure

The data is injected into watsonx Orchestrate's context under:

```
context.skills['main skill'].user_defined
```

With these fields:

`reliefweb_reports` (Array)

```
[
  {
    "title": "Medical supplies disrupted in East Africa",
    "country": "Kenya, Somalia",
    "date": "2025-11-01T12:00:00Z",
    "themes": "Logistics and Telecommunications, Health",
    "url": "https://reliefweb.int/report/...",
    "disaster": "Drought"
  }
]
```

weather_data (Object or null)

```
{
  "location": "Nairobi",
  "country": "KE",
  "temperature_celsius": 24,
  "conditions": "Clear",
  "description": "clear sky",
  "humidity_percent": 65,
  "wind_speed_kmh": 15,
  "visibility_km": 10,
  "logistics_impact": "Weather conditions favorable for logistics operations"
}
```

crisis_context (String - Summary)

```
Query: "vaccine shipment delayed"
Recent reports (3): Medical supplies disrupted...; Vaccine cold chain...; Port closure impacts...
Affected regions: Kenya, Somalia, Ethiopia
Weather conditions: Nairobi - Clear, 24°C. Weather conditions favorable for logistics operations
```

How Agents Use This Data

In IBM watsonx Orchestrate

Your agents can reference these context variables in their skills or prompts:

Example Prompt Enhancement:

You are a supply chain analyst. Use the following real-time context:

```
ReliefWeb Reports: {{context.skills['main skill'].user_defined.reliefweb_reports}}
Weather Data: {{context.skills['main skill'].user_defined.weather_data}}
Crisis Summary: {{context.skills['main skill'].user_defined.crisis_context}}
```

Analyze the user's query considering this live humanitarian data.

Agent Skill Configuration

1. **Navigate to your agent in watsonx Orchestrate**
2. **Edit the agent's system prompt or skill instructions**

3. **Add references to user_defined context variables**
4. **Test with sample queries to verify data injection**

Testing the Integration

1. Send a test message

```
"Analyze vaccine distribution delays in East Africa"
```

2. Check browser console

Look for:

```
[Chain AI] Pre-send event - enriching with live data  
[Chain AI] Injected context variables: {...}
```

3. Verify in UI

- External Situation Feed panel shows matching ReliefWeb reports
- Weather data displays for identified location

4. Agent response should reference:

- Specific crisis reports from ReliefWeb
- Current weather conditions
- Logistics impact assessment

Example Agent Responses

Without context injection:

```
"I can help analyze vaccine distribution. What specific region are you concerned about?"
```

With context injection:

```
"Based on the latest ReliefWeb reports, vaccine distribution in Kenya and Somalia is currently impacted by port closures and transportation delays. Current weather in Nairobi shows favorable conditions (24°C, clear skies), so weather is not a contributing factor. The primary issues are logistical - I recommend focusing on alternative route activation and emergency air transport as outlined in recent supply chain disruption reports."
```

Customization

Modify Context Structure

Edit `src/components/WatsonXChat.tsx` in the `pre:send` event handler:

```

event.data.context = {
  ...event.data.context,
  skills: {
    'main skill': {
      user_defined: {
        // Add your custom fields here
        reliefweb_reports: contextData.reports,
        weather_data: contextData.weather,
        crisis_context: contextData.summary,
        custom_field: 'your value'
      }
    }
  }
};

```

Change Data Fetch Behavior

Modify `fetchLiveDataForContext()` to:

- Change number of reports fetched
- Filter by specific themes/disasters
- Add additional API sources
- Customize weather criteria

Troubleshooting

Context not appearing in agent responses

1. Check console for injection logs

- Should see "Injected context variables" message

2. Verify agent prompt references context

- Agent must explicitly use `{{context.skills['main skill'].user_defined.*}}`

3. Confirm pre:send event fires

- Some embed configurations may not support all events

Weather data null

- Location couldn't be inferred from ReliefWeb reports
- OpenWeather API key missing or invalid
- Country name not in capital city mapping

ReliefWeb reports empty

- Query too specific with no matching reports
- Fallback to recent reports should still provide data

- Check network tab for API errors

Environment Variables

Required in `.env.local`:

```
VITE_OPENWEATHER_API_KEY=your_key_here
```

Optional (for authentication):

```
VITE_WXO_SECURITY_DISABLED=true  
# OR  
VITE_WXO_USE_IAM=true  
VITE_WATSONX_API_KEY=your_key_here
```

API Rate Limits

- **ReliefWeb**: No authentication required, reasonable rate limits
- **OpenWeather**: 60 calls/minute (Free tier), 1000 calls/day

To avoid rate limits:

- Data is fetched only when user sends a message
- Weather lookups cached briefly in UI state
- Consider implementing request throttling for high-traffic scenarios

Next Steps

1. **Configure your agents** to use the injected context variables
2. **Test with real humanitarian scenarios** to validate data relevance
3. **Monitor agent responses** to ensure context is being utilized
4. **Customize data structure** based on your agent's needs
5. **Add additional APIs** as needed for specialized analysis

Support

- Check browser console for detailed logs prefixed with `[Chain AI]`
- Verify API keys are correctly set in `.env.local`
- Ensure watsonx agents are published and accessible
- Review IBM watsonx Orchestrate documentation for context variable usage